BSSE System and Software Engineering

SCISYS

FAST II

Automatic Source-code-based Testing, Improvement

Final Presentation

Noordwijk, *date tbd*

ESA Contract No. 4000116014 (GSTP)

BSSE Team:	Rainer Gerlich, Ralf Gerlich
SCISYS Team:	Allan Pascoe, Glenn Johnson
ESA PO:	Maria Hernek

Dr. Rainer Gerlich
Auf dem Ruhbuehl 181
88090 Immenstaad
Germany

Tel. +49/7545/91.12.58
Fax +49/7545/91.12.40
Mobile +49/171/80.20.659
email Rainer.Gerlich@bsse.biz

BSSE System and Software Engineering

Contents

SCISYS

- **The Goals of the Project**
- **The FAST Approach**
- **Open Tool Interface**
  - VectorCAST Interface
  - Cantata Interface
- **Requirements-Based Testing**
- **Benchmarking**
- **Lessons Learned**
- **Conclusions and Outlook**

The FAST Approach

About FAST and DCRTT

- **FAST**
  - Flow-Optimised Automated Source-code-based Testing
  - automate the test process from test data generation to report generation

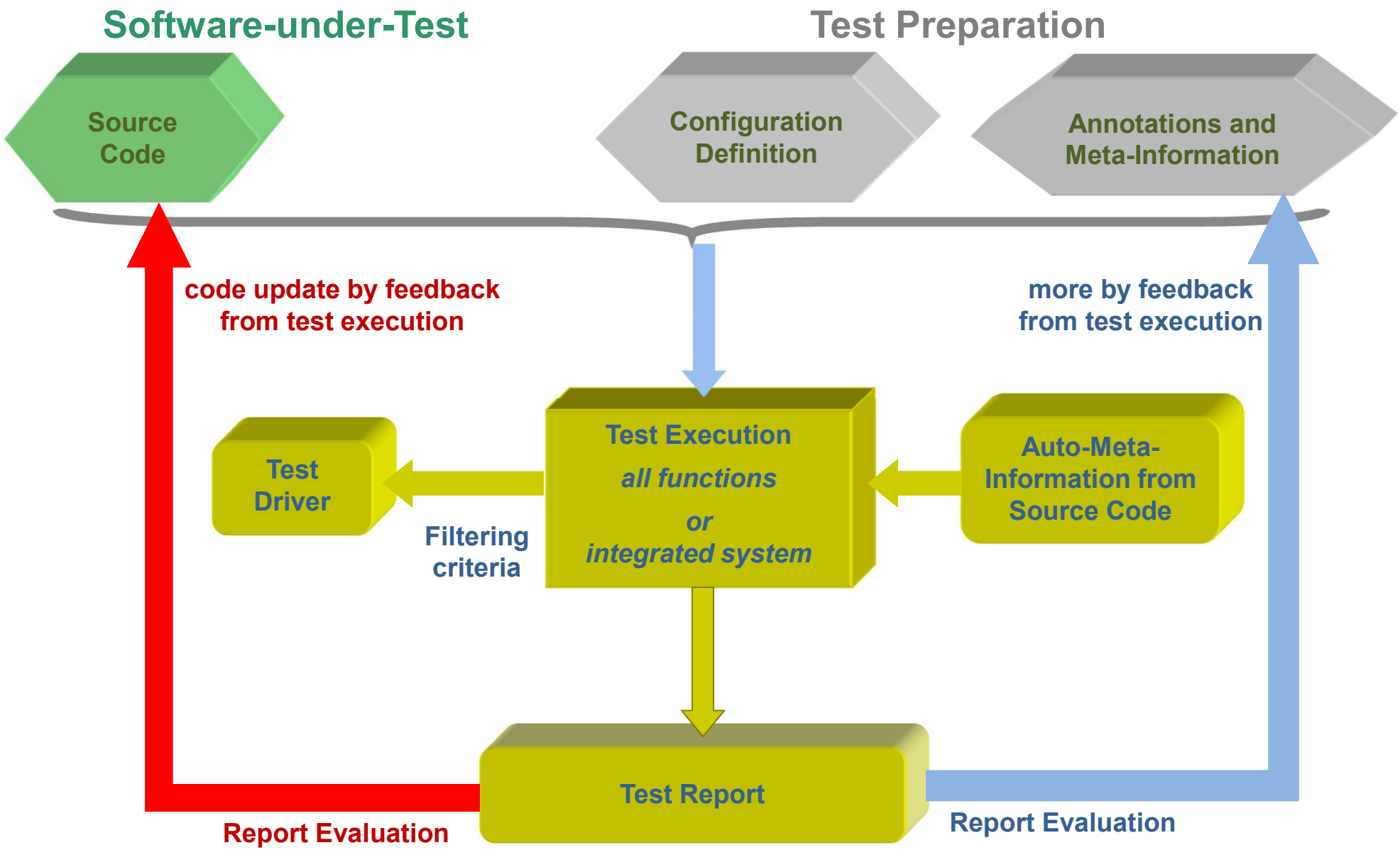
- **DCRTT**
  - Dynamic C Random Test Tool
  - following DARTT, Dynamic Ada Random Test Tool
  - tool supporting the FAST approach

Goals of the Project

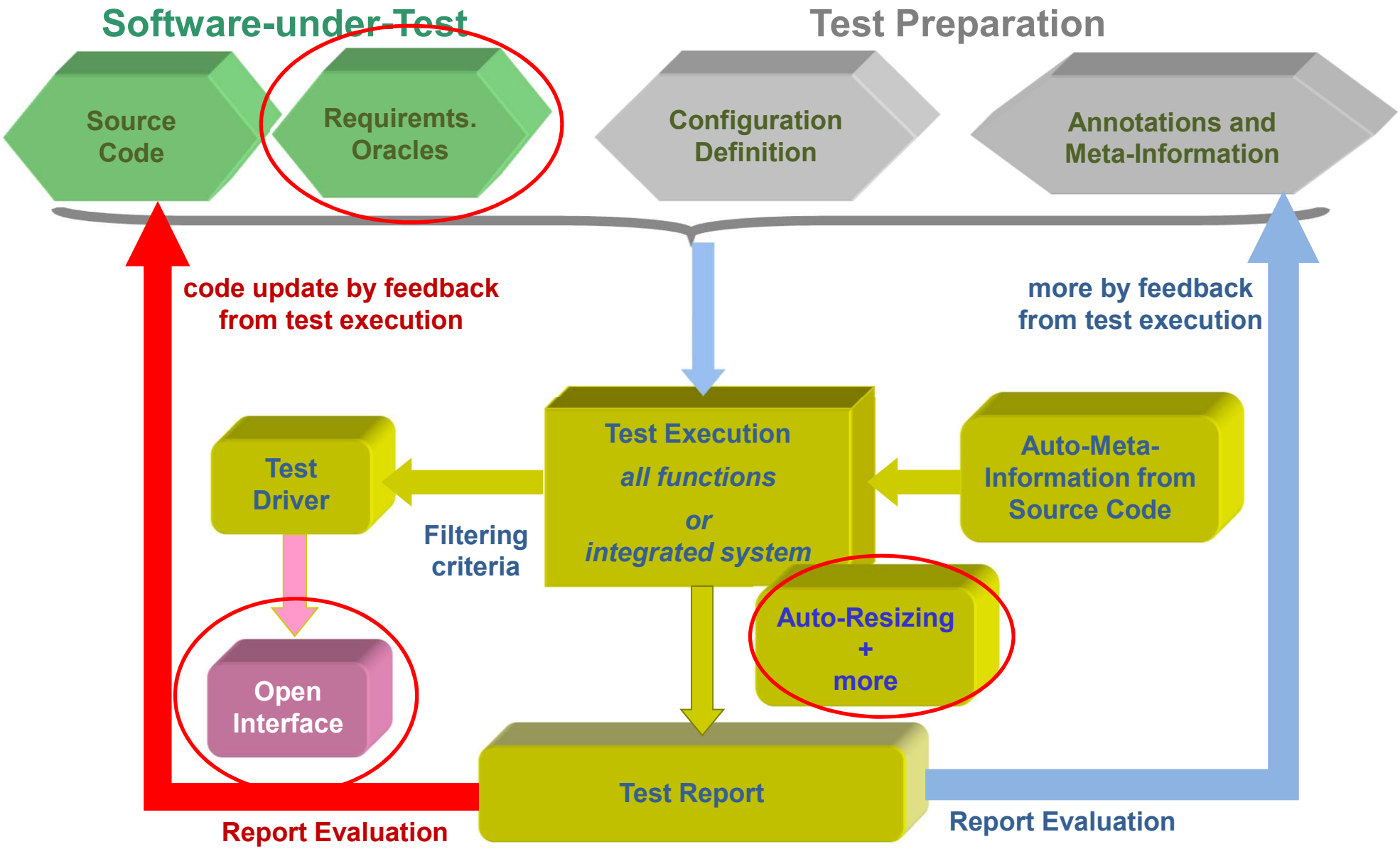
- **Identify DCRTT improvements**
  - establish a list of improvements during the previous activities
  - define priorities for implementation
- **Define an open interface and link to other tools**
  - support export of auto-generated test vectors
  - consider Cantata and VectorCAST as certified tools
- **Analyse support of Requirements-Based Testing (RQBT)**
  - investigate how the gap from auto-generated test vectors to requirements can be closed
  - analyse requirements of a typical space project
  - define a concept for implementation
- **Assess the achievements by Benchmarking**
  - perform benchmarking with other static analysers
  - support an assessment for TRL 5
  - perform a demo

The FAST Approach

From Test Preparation to Result Evaluation *before FASTII*

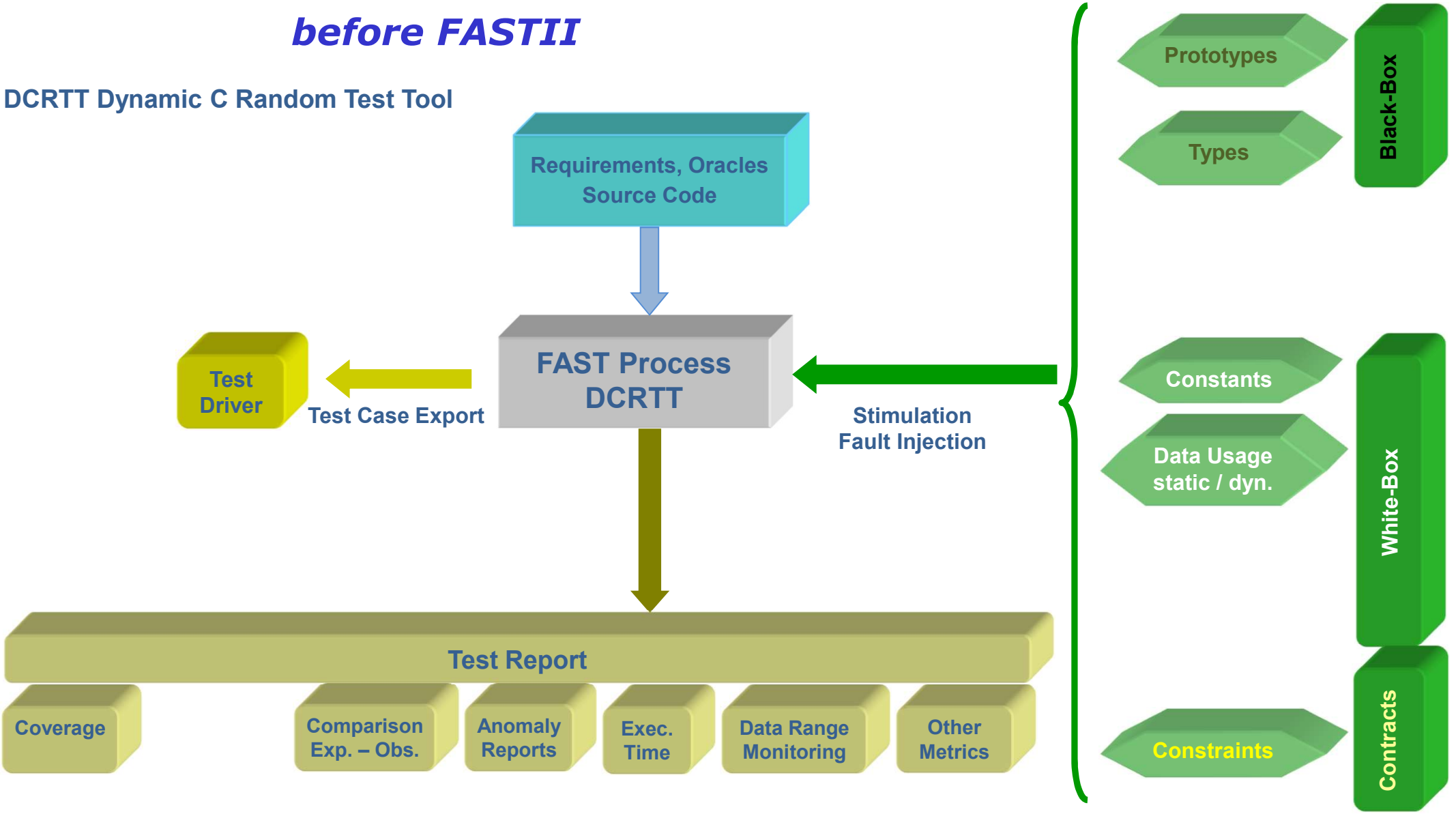


From Test Preparation to Result Evaluation *at the end of FASTII*



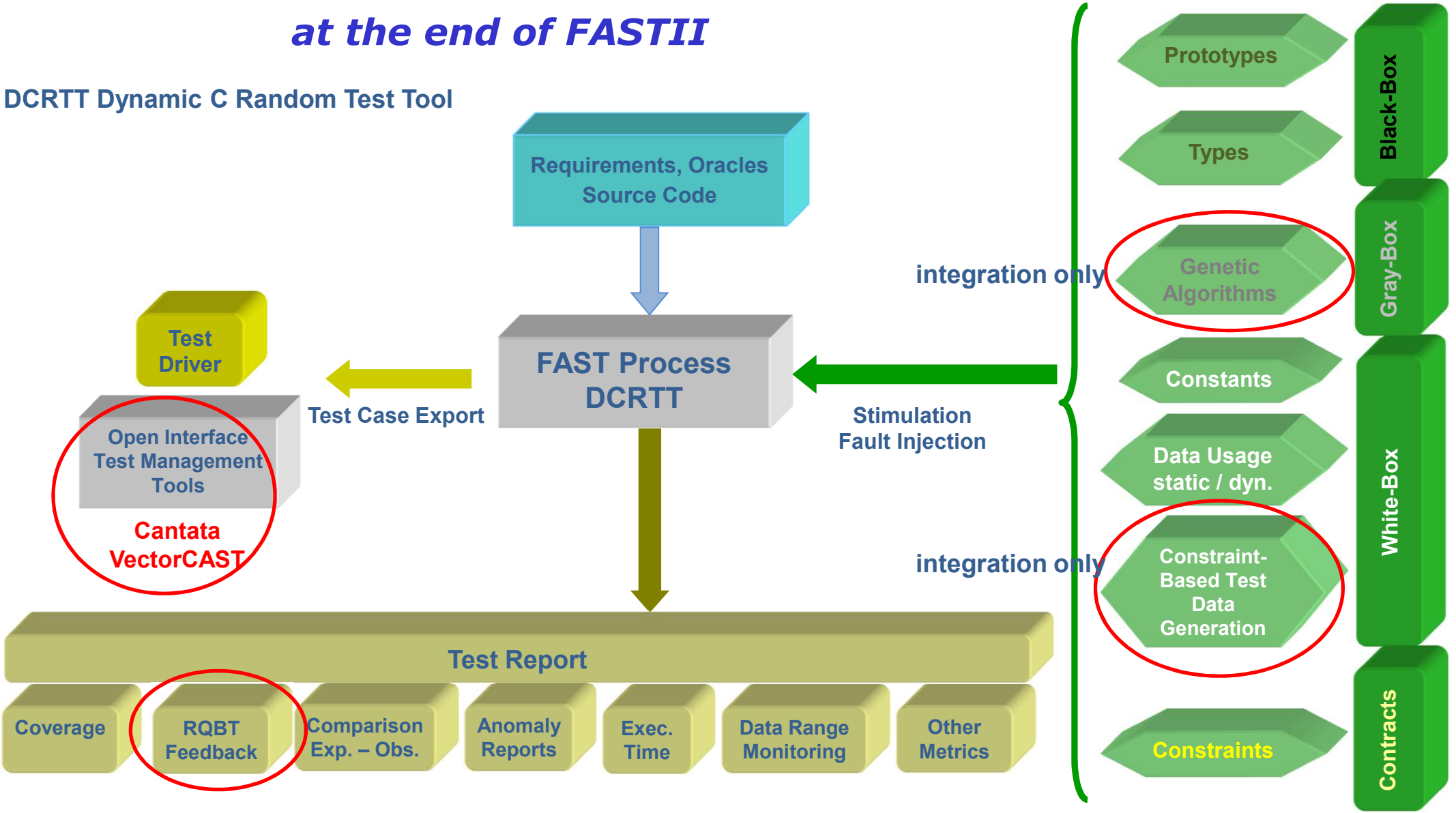
Interfaces of the FAST Test Process in DCRTT

before FASTII

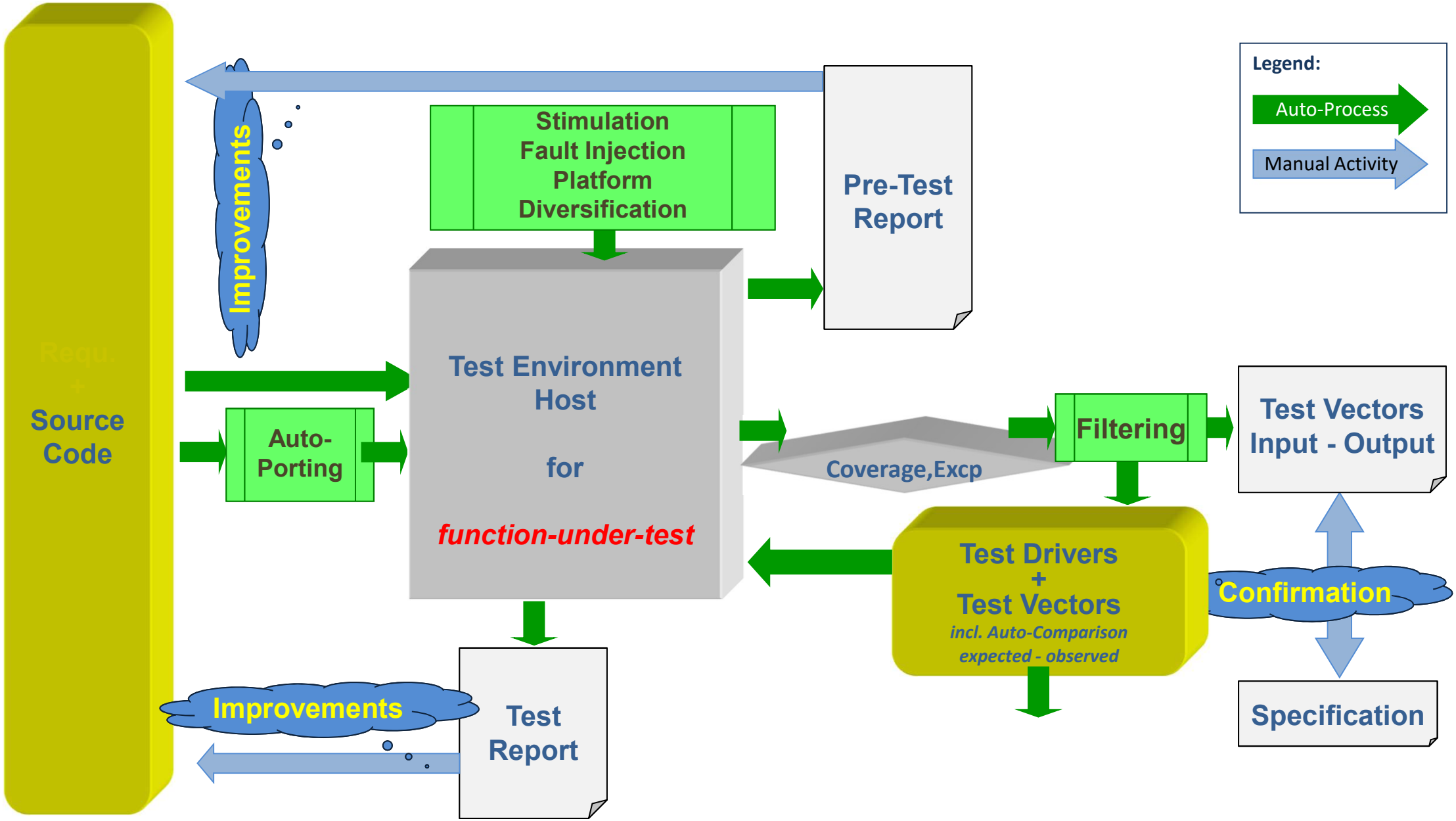


Interfaces of the FAST Test Process in DCRTT

at the end of FASTII



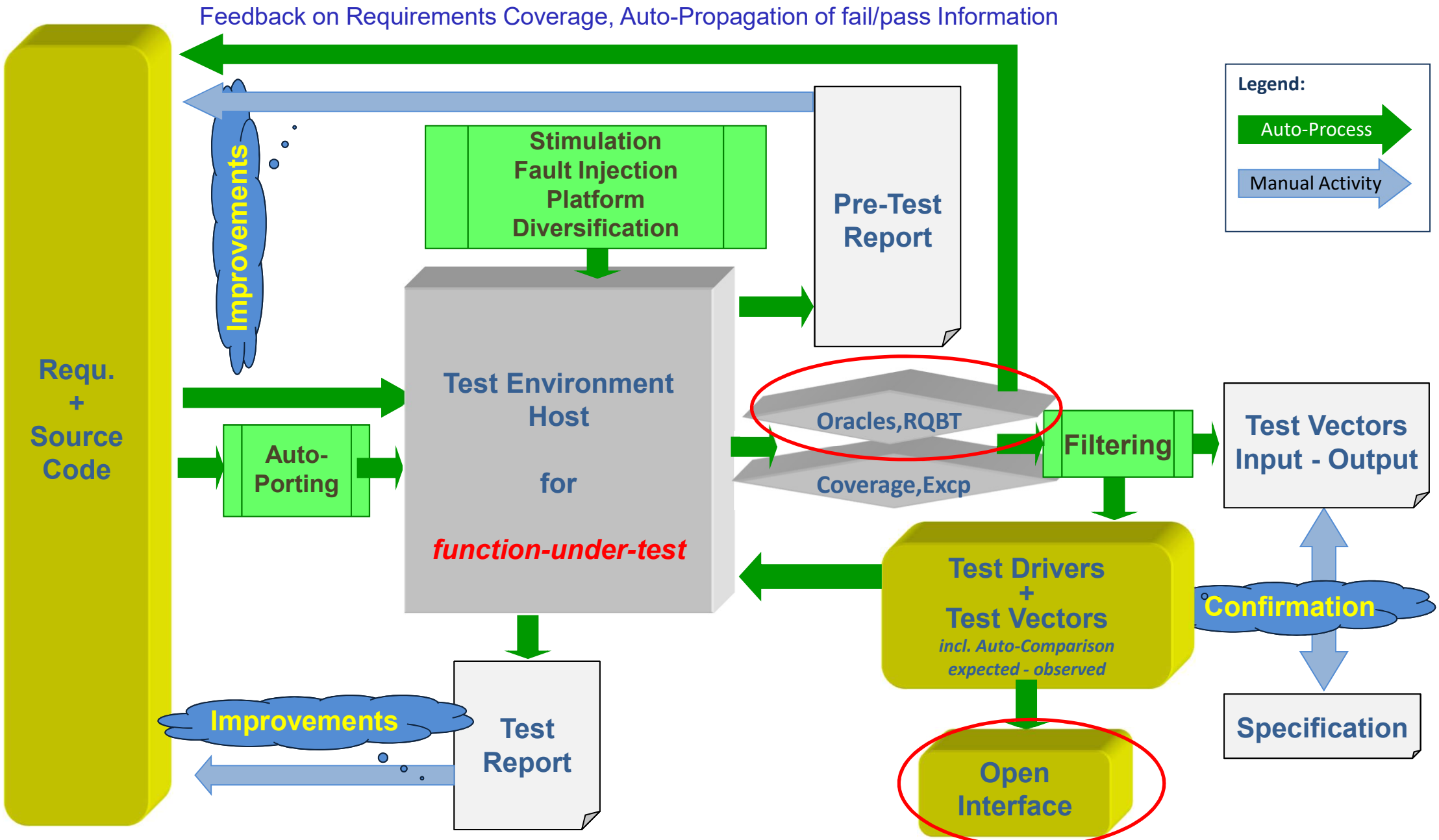
The FAST Test Process Detailed Flow *before FASTII*



The FAST Test Process Detailed Flow *at the end of FASTII*

BSSE System and Software Engineering

SCISYS



List of Major Extensions

- **Improvements**
  - support csv-Format for defect reporting
  - auto-resizing of pointer parameters
  - auto-resizing of unconstrained arrays in parameter list
  - *support of assertions*
  - *address validation*

  Implementation before FASTII
  - *support of object size validation for C library routines (memcpy, …)*
  - *constrained-based test data generation*
  - *genetic algorithm for untyped byte streams (telecommands)*
- **Open Tool Interface**
  - link to Cantata
  - link to VectorCAST
- **RQBT**
  - support of oracles derived from requirements
  - support of bottom-up propagation of requirements coverage and passed/failed results

Resizing Example

```
char *ptr=malloc(???)
char *ucArr      [???];
myFunc(ptr,ucArr);
```

Test environment, auto-generated

Index may be defined at run-time,
correlation with maximum may not be possible

If too few elements are allocated ⇨ false positives

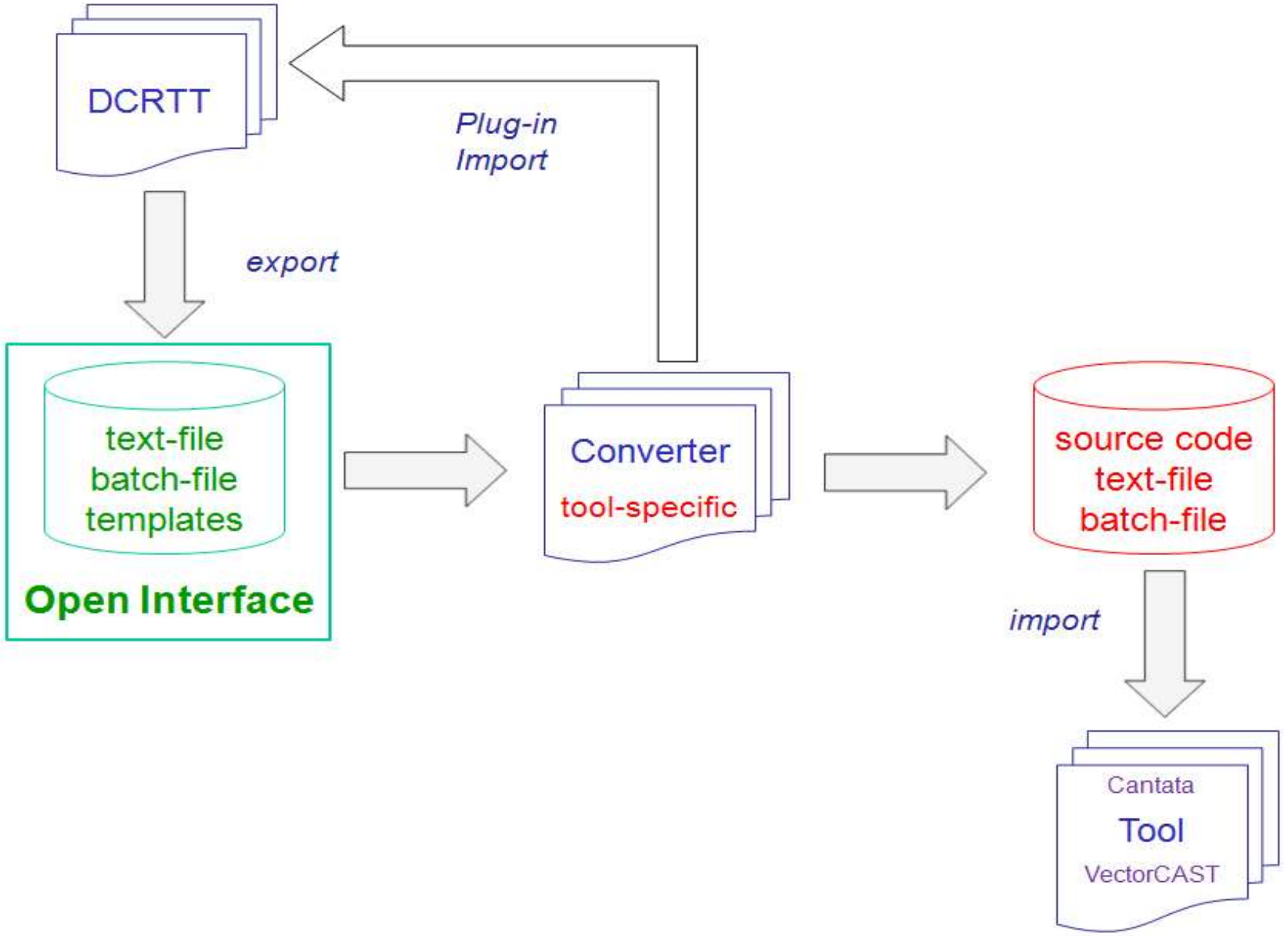
⇨ Resize dynamically, record resizing and check against RQ

```
static unsigned int idx1=0,idx2=0;
int myFunc(char *ptr, char *ucArr[])
    myFunc2(ptr,ucArr);
    int myFunc2(char *para1, char *para2[]);
        para1[idx1]=1;
        para2[idx2]=2;
```

Software-under-Test

Open Tool Interface

Open Tool Interface Principal Approach



VectorCAST

Output VectorCAST Coverage Report

Aggregate Coverage Report

Configuration Data

Date of Report Creation:	25 MAY 2018
Time of Report Creation:	11:51:10 PM

Aggregate Coverage

Code Coverage for Unit: hello_vcast_1.c'1

```
-- Coverage Type: Statement+MC/DC
-- Unit: hello_vcast_1
-- Test Case: Aggregate
                #include "C:/PROJEKTE/DCRTT/MYTEST/TESTSRCDOIF_VC_ALL_1_5_ALL/DCRTT_test_addons.h"
                #include "C:/PROJEKTE/DCRTT/SRC/DCRTT_ENV/DCRTT_TS/inout_def.h"
                extern int DCRTT_stdout_fprintf(const char *form, ...);
                extern void *missFunc(int para1, int *ptr);
                extern int *missData;
                int *globArrNULL =((void *)0);
                int *globArrNonInit;
                int checkOORdown(int *arr)
                {
1 0      (T)     checkOORdown
1 1      *         int ii,ret=0;
1 2      (T)(F)    for (ii=50;
1 2.1    (T)(F)    ii>=-1;ii--)
1 3      *           ret+=arr[ii];
1 4      *         return ret;
                }
                int checkOORup(int *arr)
                {
2 0      (T)     checkOORup
2 1      *         int ii,ret=0;
2 2      (T)(F)    for (ii=0;
2 2.1    (T)(F)    ii<=15;ii++)
2 3      *           ret+=arr[ii];
2 4      *         return ret;
                }
                int *checkPtrReturn(int *arr)
                {
3 0      (T)     checkPtrReturn
3 1      *         int ii,ret=0;
3 2      (T)(F)    for (ii=0;
3 2.1    (T)(F)    ii<=25;ii++)
3 3      *           ret+=arr[ii];
3 4      *         return arr;
                }
```

Example VectorCast Test Script (2) unconstrained array

```
-- Test Case Script
-- Environment        :
vc_test_hello_vcast_1_BSSE_main_7
-- Function Under Test: hello_vcast_1 -
BSSE_main 7 of hello_vcast_1.c
-- Script Features
TEST.SCRIPT_FEATURE:C_DIRECT_ARRAY_INDEXING
TEST.SCRIPT_FEATURE:CPP_CLASS_OBJECT_REVISION
TEST.SCRIPT_FEATURE:MULTIPLE_UUT_SUPPORT
TEST.SCRIPT_FEATURE:STANDARD_SPACING_R2
TEST.SCRIPT_FEATURE:OVERLOADED_CONST_SUPPORT
-- End of header hello_vcast_1 - BSSE_main 7 of
hello_vcast_1.c
--  vc_test_7.tst generated by
dcrtt_open_if_cnv_vc.c for tool vc on <date>
-- Test File: hello_vcast_1.c
TEST.UNIT:hello_vcast_1
TEST.SUBPROGRAM:BSSE_main
-- mangled name BSSE_main
-- List of relevant data of function BSSE_main
-- #tot    para=  2
-- #func   para=  2
-- #glob   para=  0
-- #constr para=  0
-- Parameters
--  signed int  argc
--  char * argv[UC_LIT2]
-- Return
--  signed int _return_
TEST.NEW
TEST.NAME:(CL)BSSE_main.001
-- derived from DCRTT test case     1
TEST.NOTES:
No requirements provided
TEST.END_NOTES:
TEST.FLOATING_POINT_TOLERANCE: 9.999999747378751635555e-06
TEST.VALUE:hello_vcast_1.BSSE_main.argv[0]:<<malloc 26>>
TEST.VALUE:hello_vcast_1.BSSE_main.argv[1]:<<malloc 26>>
TEST.VALUE:hello_vcast_1.BSSE_main.argv[2]:<<malloc 26>>
TEST.VALUE:hello_vcast_1.BSSE_main.argv[3]:<<malloc 26>>
TEST.VALUE:hello_vcast_1.BSSE_main.argv[4]:<<malloc 26>>
TEST.VALUE:hello_vcast_1.BSSE_main.argc: -2147483648
TEST.VALUE:hello_vcast_1.BSSE_main.argv[0]: ""
TEST.VALUE:hello_vcast_1.BSSE_main.argv[1]: "lxivmf{lurnmkdzwlqrr
rjqg"
TEST.VALUE:hello_vcast_1.BSSE_main.argv[2]: "g
uxxclxgjnorgwhuqouzjmgi"
TEST.VALUE:hello_vcast_1.BSSE_main.argv[3]: "LQWT7WNaMA0K0HKJTMR
D1423"
TEST.VALUE:hello_vcast_1.BSSE_main.argv[4]: "1@i5}A6}7\\'%kB^I$2r
2)7m3"
TEST.VALUE:hello_vcast_1.BSSE_main.return: -2147483648
TEST.EXPECTED_USER_CODE:hello_vcast_1.BSSE_main.argc
{{ (signed long)<<hello_vcast_1.BSSE_main.argc>> == ( (signed
long)-2147483648) }}
TEST.END_EXPECTED_USER_CODE:
TEST.EXPECTED_USER_CODE:hello_vcast_1.BSSE_main.argv
{{ strcmp( <<hello_vcast_1.BSSE_main.argv>>[0] ,  "1234567890" ) }}
{{ strcmp( <<hello_vcast_1.BSSE_main.argv>>[1] ,  "1234567890" ) }}
{{ strcmp( <<hello_vcast_1.BSSE_main.argv>>[2] ,  "1234567890" ) }}
{{ strcmp( <<hello_vcast_1.BSSE_main.argv>>[3] ,  "1234567890" ) }}
{{ strcmp( <<hello_vcast_1.BSSE_main.argv>>[4] ,  "1234567890" ) }}
TEST.END_EXPECTED_USER_CODE:
TEST.EXPECTED_USER_CODE:hello_vcast_1.BSSE_main.return
{{ (signed long)<<hello_vcast_1.BSSE_main.return>> == ( (signed
long)0) }}
TEST.END_EXPECTED_USER_CODE:
TEST.END
```

Cantata

BSSE System and Software Engineering

Output Cantata for Test Script / Statement Coverage

SCISYS

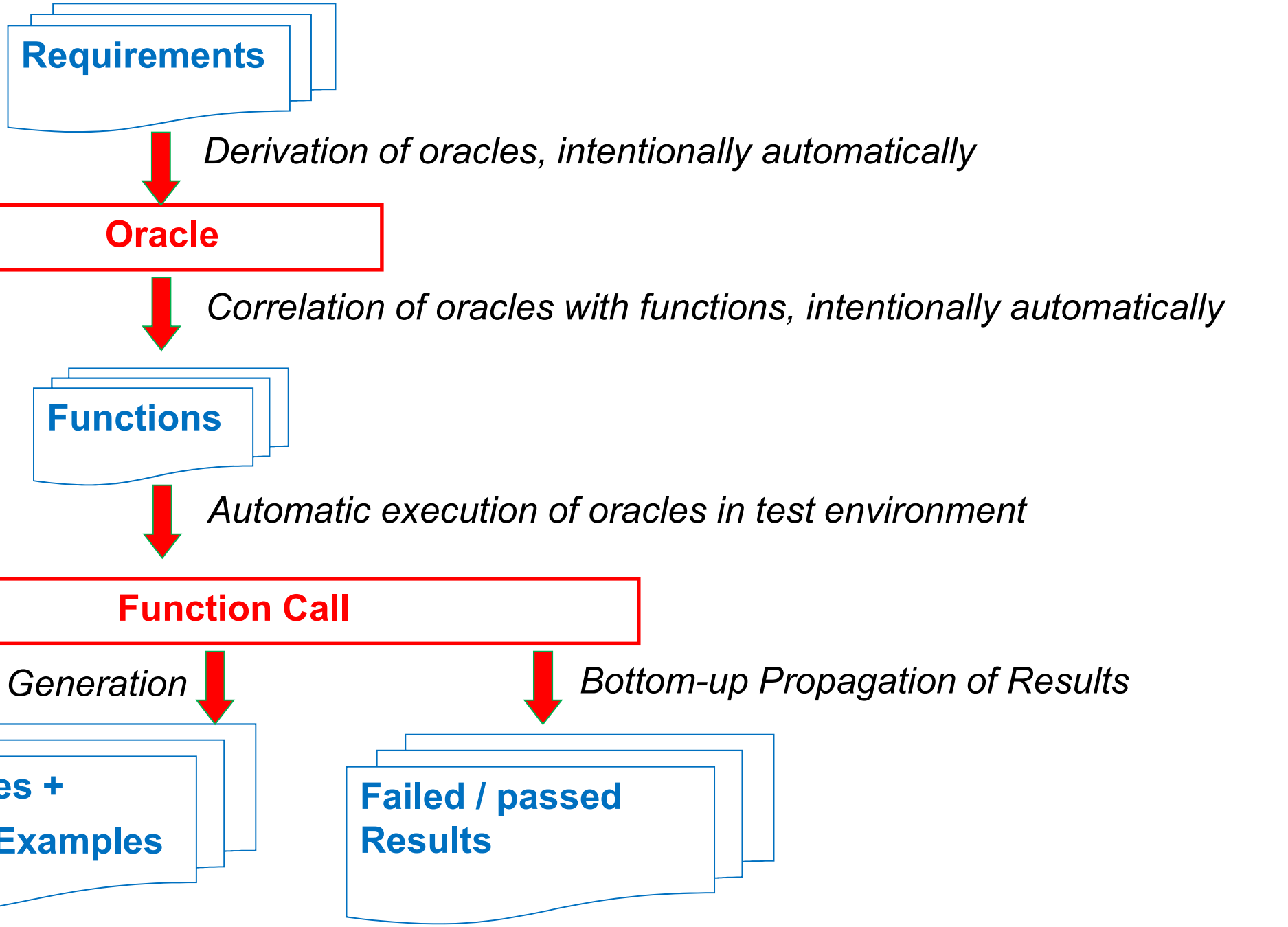
```
Using Cantata gui, Log Level defaulted to detailed.
IMPORT_COVERAGE: coverage data read from "C:\Projekte\dcrtt\mytest\testsrcdoif_
cantpp12_all_files\\cantpp_host\DCRTT_CANTPP_PROJECT_host\Cantata\cantpp_cover_
overall.covg" accumulated with current
-------------------------------------------------------------------------------
Coverage results for  statement, basic block, decision, masking effectiveness,
relational, loop,
                for "*" in context "*"
                with executed and un-executed details
                including infeasible, including catch-clauses
-------------------------------------------------------------------------------
hello_cantpp_1.c(152):BSSE_main(int ,char **)
statement coverage details (with executed and un-executed cases)

hello_cantpp_1.c(155):      stmnt   1 (other)                          1024
hello_cantpp_1.c(156):      stmnt   2 (loop)                           1024
hello_cantpp_1.c(156):      stmnt   3 (loop)                          31744
hello_cantpp_1.c(158):      stmnt   4 (cond)                          31744
hello_cantpp_1.c(159):      stmnt   5 (other)                          6144
hello_cantpp_1.c(161):      stmnt   6 (other)                         25600
hello_cantpp_1.c(163):      stmnt   7 (return)                         1024

"BSSE_main"                                          executed             7
"BSSE_main"                                        un-executed            0

hello_cantpp_1.c(152):BSSE_main(int ,char **)
basic block coverage details (with executed and un-executed cases)

hello_cantpp_1.c(153):      block   1                                  1024
hello_cantpp_1.c(157):      block   2                                 31744
hello_cantpp_1.c(159):      block   3                                  6144
hello_cantpp_1.c(161):      block   4                                 25600

"BSSE_main"                                          executed             4
"BSSE_main"                                        un-executed            0
```

Output Cantata for Test Script / Decision Coverage

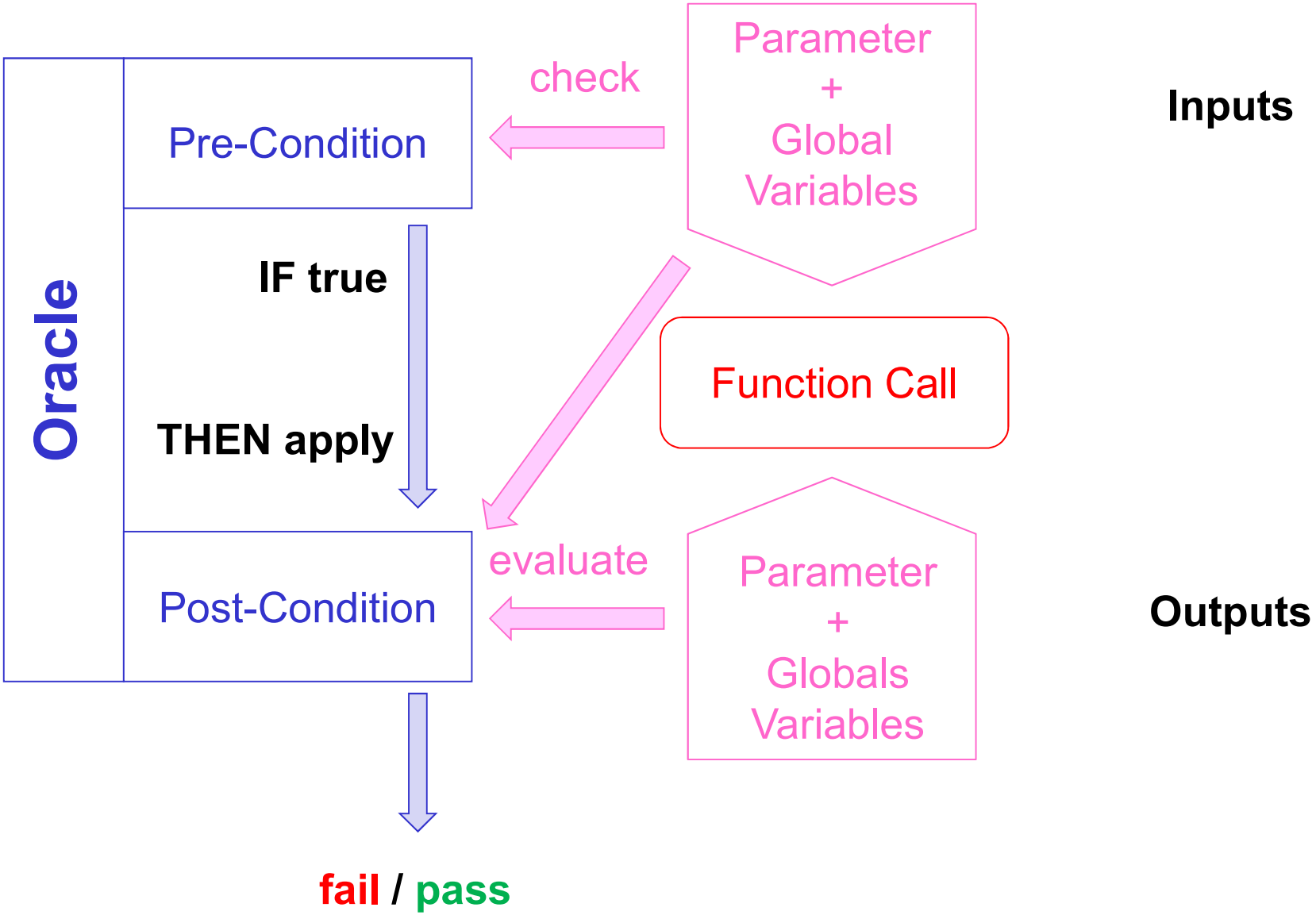
```
hello_cantpp_1.c(152):BSSE_main(int ,char **)
decision coverage details (with executed and un-executed cases)

hello_cantpp_1.c(156):       decn   1 (for)    branch TRUE              31744
hello_cantpp_1.c(156):       decn   1 (for)    branch FALSE              1024
hello_cantpp_1.c(158):       decn   2 (if)     branch TRUE               6144
hello_cantpp_1.c(160):       decn   2 (if)     branch FALSE             25600

"BSSE_main"                                        executed                4
"BSSE_main"                                      un-executed               0

hello_cantpp_1.c(166):BSSE_main2(int ,char **)
statement coverage details (with executed and un-executed cases)

hello_cantpp_1.c(169):      stmnt   1 (other)                            512
hello_cantpp_1.c(170):      stmnt   2 (loop)                             512
hello_cantpp_1.c(170):      stmnt   3 (loop)                           16384
hello_cantpp_1.c(172):      stmnt   4 (cond)                           16384
hello_cantpp_1.c(173):      stmnt   5 (other)                          13312
hello_cantpp_1.c(175):      stmnt   6 (other)                           3072
hello_cantpp_1.c(177):      stmnt   7 (return)                           512

"BSSE_main2"                                       executed                7
"BSSE_main2"                                     un-executed               0

hello_cantpp_1.c(166):BSSE_main2(int ,char **)
basic block coverage details (with executed and un-executed cases)

hello_cantpp_1.c(167):      block   1                                    512
hello_cantpp_1.c(171):      block   2                                  16384
hello_cantpp_1.c(173):      block   3                                  13312
hello_cantpp_1.c(175):      block   4                                   3072
```

Requirements-Based Testing (RQBT)

Requirements and Coverage



Structure of an Oracle



Oracle Examples

Three Oracles

status==active && mode==mode1 ? moniFlag==true

status==active && mode==mode2 ? moniFlag==true

status==active && mode==mode3 ? moniFlag==false

Pre-condition:
if true check post-condition

Post-condition:
if true: pass
if false: fail

Function	Requirement	Oracle		Number of Tests	Oracle Output			RQ fully covered	RQ verified
		Pre-Condition	Post-Condition		Coverage	true	false		
x*x	∀x∈{double} $\sqrt{x^2}$ shall not differ from x more than ε	x≤-1.0 || x≥1.0	(fabs(fabs(x)-sqrt(retVal))/x)<eps	302	299	225	74	yes	no
		x>-1.0 || x<1.0	fabs(fabs(x)-sqrt(retVal))<eps		3	3	0		
abs(x)	∀x∈{sint} abs(x) shall be ≥0	RQBT_FORALL(x)	retVal>=0	302	302	301	1	yes	no

Conclusion on Requirements Analysis

- **Readiness for Auto-Extraction of Information**
  - only some requirements found suitable for auto-generation of oracles
  - machine-interpretable requirements required
  - guidelines required
- **Requirements Top-Down Tracking**
  - continuous chain of tracking required
  - lack of tracking information down to function level
- **DCRTT Implementation**
  - text notation for oracles defined
  - infrastructure available supporting this notation
  - support for bottom-up propagation available
- **RQBT Demo**
  - manually defined oracle examples
  - some related to suitable requirements of space application
  - some defined for test and demonstration

Benchmarking

Selected Tools

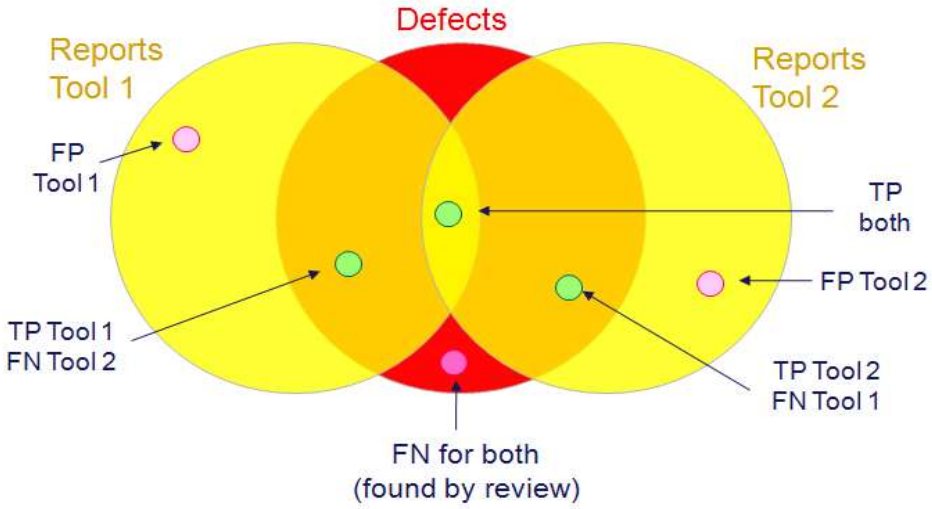
	Analysis Type	Analysis Approach	Soundness
DCRTT	dynamic	test, auto-stimulation and auto-test data generation	not sound
Astree	static	abstract interpretation	sound
CodeProver			sound
BugFinder			not sound
QA/C	static	symbolic execution, dataflow analysis	not sound

Remarks on Benchmarking Evaluation Results

- **Boundary Conditions**
  - Benchmarking was performed at the end of development
  - Many reports issued by the tools (up to 30.000)
  - Unclear: Number of reports and effort in case of continued application
- **Application Impact**
  - number and type of reports strongly application dependent
- **Configuration Impact**
  - number and type of reports strongly (tool-)configuration dependent
- **Report Selection Impact**
  - evaluation results strongly depend on selection process
  - number of reports issued may heavily differ between tools

Report Classification

		Actual Condition	
		Defect present	Defect not present
Reported Condition	Defect present	True Positive	False Positive
	Defect not present	False Negative	True Negative



Classification Category	Criterion	Applied Condition
Validity	tool	Is the tool message formally correct?
	state	Can an undesired state be reached?
Context	with context	The execution conditions may be constrained by the calling function (caller)
	without context	The execution conditions are not constrained

The Application

- **Two versions**
  - early version with potentially more defects
  - late version with potentially less defects
- **Intention of using two versions**
  - evaluate impact on reporting by number of reports
  - no significant difference found

Version	h-Files	c-Files	All Files	KLOC - h	KLOC - c	KLOC - All	Functions
early	170	150	320	32	151	183	3200
late	170	150	320	29	167	196	3400

Figures approx.

Item	Version	
	early	late
max. type nesting level	9	9
missing functions (OS interface, assembler), stubs	117	121
DCRTT support functions for generation of output	368	369
missing data	4	4

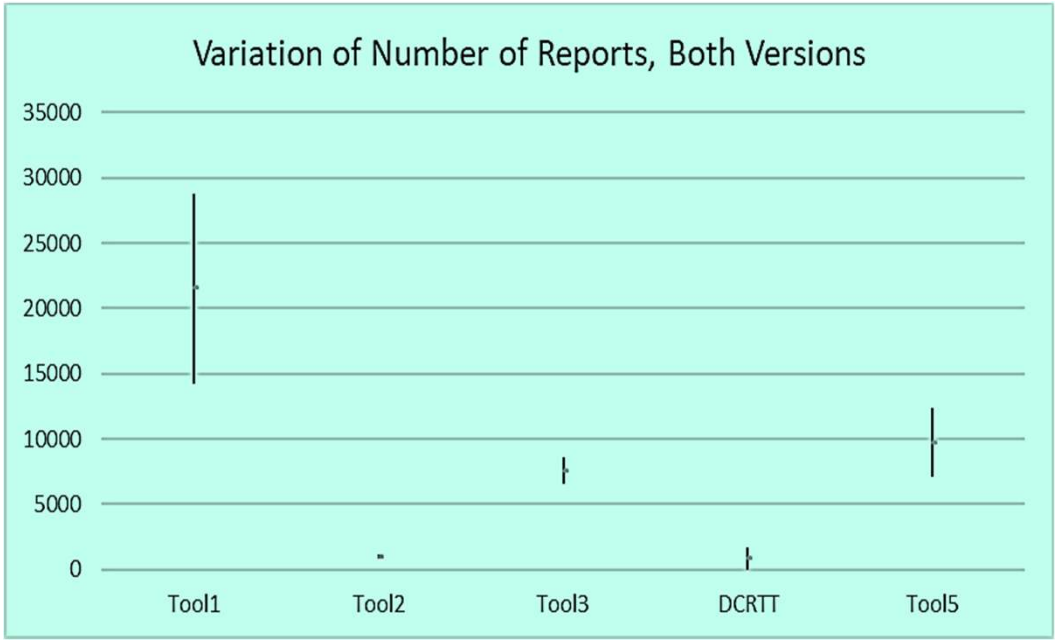
Task Types
periodic
synchronous
sporadic
standard

Analysis and Test Modes

Analysis Mode			Tool		
	DCRTT	QAC	Astree	BugFinder	CodeProver
EM 1	x		x	x	x
EM 2	x		x	x	x
EM 3			x	x	x
Unit testing	x				
functionwise		x			

Execution Mode	Description
EM 1	deterministic / sequential execution of the task bodies
EM 2	non-deterministic / random execution of the task bodies
EM 3	modelling of concurrent execution of task bodies with pre-emption
Unit testing	every function is subject of stimulation / testing
functionwise	every function is independently analysed

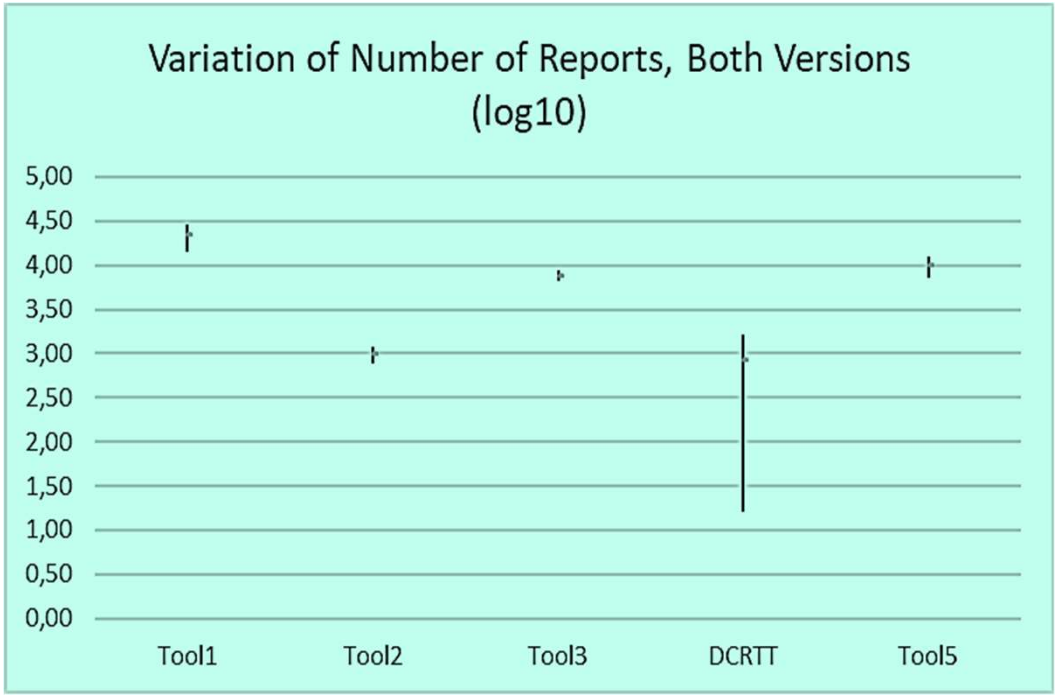
Variation of Report Figures Both Versions



To be considered:

different reporting policies of tools

- reporting for every location or not
- reporting for every path or not
- duplication of reports or not



Variation of Report Figures Examples

Early Version						
Defect Type	Tool1	Tool2	Tool3	DCRTT	Tool5	Comment
Assert	243	6	251	219	0	compromised by stubbing
Concurrency Issues	10044	741	2014	n/a	n/a	non-relevant due to non-representative scheduling
Unused Result	4257	701	0	n/a	0	
Uninitialized Variable	1521	36	1298	n/a	19	

Late Version						
Defect Type	Tool1	Tool2	Tool3	DCRTT	Tool5	Comment
Assert	363	6	343	227	0	compromised by stubbing
Concurrency Issues	10755	807	2633	n/a	n/a	non-relevant due to non-representative scheduling
Unused Result	4909	732	0	n/a	0	
Uninitialized Variable	1140	26	1215	n/a	4	

Analysis Base

- **Entry-Point Function**
  - Astree only supports analysis from single entry-point function
  - auto-construction of entry-point function, the same used for all static analysers
  - calls all task bodies
- **Analysis Modes (Invocation of Task Bodies)**
  - deterministic, grouped by task types, fixed sequence (mode 1)
  - non-deterministic, random call of the sequence (mode 2)
  - multi-tasking, using pre-emptive support if supported (Astree, CodeProver) (mode 3)
- **DCRTT Modes**
  - unit testing: test environment built for every function of the application
  - additionally: deterministic run of single entry point
  - In both cases: Instrumentation for coverage and fault detection

Report Consolidation

- **Report Contents**
  - text differs widely between tools for the same defect type
  - multiple reports with different text may be issued for the same defect
  - if more than one report is provided for a line or statement
    - Use column for distinction (may differ between tools)
    - statement id
    - parameter or index id
- **Automated merge of all reports from all tools**
  - mapping scheme required to make reports comparable (consolidation)
  - standard defect types applied
  - tool specific output format converted into standard report format
  - no consideration of column or other id:
    most important is the message per location (file, line), no TP locations are lost
- **Consolidated Output**
  - merged list of reports according to standard defect type and location
  - list of defects per location for cross-checking of mapping

Standard Defect Types (1/2)

Id	Standard Defect Type	Criticality
1	Array Index Out-of-Bounds	Critical
2	Assert failure	Critical
3	Dangling Pointer	Critical
4	Dereference of Invalid or NULL Pointer	Critical
5	File Access Error	Critical
6	Format String Mismatch	Critical
7	Invalid arithmetic operation	Critical
8	Invalid function pointer	Critical
9	Invalid Return Statement	Critical
10	Loss of Precision	Critical
11	Macro Use with Unintended Consequences	Critical
12	Non-terminating Loop	Critical
13	Possible Invalid Use of Function	Critical
14	Possible Recursion	Critical
15	Resource Leak	Critical
16	Undefined Result	Critical
17	Uninitialized Variable	Critical
18	Unintended Use of Implicit Member Function	Critical
19	Arithmetic Operation on NULL Pointer	Warning
20	Arithmetic Overflow	Warning
21	Cast to pointer of incompatible types	Warning
22	Comparison of floating-point values	Warning
23	Concurrency Issues	Warning
24	Conflicting Declarations	Warning
25	Incomplete List of Cases for enum-Type without default	Warning
26	Intended Change of Invariant Data	Warning

Criticality

⇨ hint that a defect type may turn out as

highly critical
less critical
uncritical
to be ignored
(remarks, comments,…)

⇨ recommendation what should be checked first.

⇨ used for statistics

Standard Defect Types (2/2)

Id	Standard Defect Type	Criticality
27	Invalid pointer operation	Warning
28	Invariant Condition	Warning
29	Invariant Expression	Warning
30	Loss of Update	Warning
31	Memory Size Mismatch	Warning
32	Name overloading	Warning
33	Parameter Type Mismatch in Function Call	Warning
34	Possible invalid arithmetic operation	Warning
35	Possible invalid pointer operation	Warning
36	Possible Loss of precision	Warning
37	Possible misuse of signed integer	Warning
38	Tainted Data	Warning
39	Timeout during Execution	Warning
40	Unnecessary Loop Construct	Warning
41	Unnecessary Operation	Warning
42	Unreachable Code	Warning
43	Unreliable arithmetic cast	Warning
44	Unreliable pointer cast	Warning
45	Unused Result	Warning
46	Change of Data expected, but missing	Uncritical
47	Incomplete List of Cases for enum-Type with default	Uncritical
48	Inconsistent Overloading	Uncritical
49	Multiple return paths	Uncritical
50	Security Issue	Uncritical
51	Unintended Change of Data	Uncritical
52	DefectTypeIgnore	ignore

BSSE System and Software Engineering

Defect Type Support by DCRTT

SCISYS

Standard Defect Type	Critic.	DCRTT Messages	Description
Array Index Out-of-Bounds	C	CorrMem	Corrupted memory detected
		OutOfRangeLow	Index <0
		OutOfRangeHigh	Index > maximum value for constrained arrays
Assert failure		AssertFailed	Assertion failed
Dereference of Invalid or NULL Pointer		*Excp	A number of different messages on exceptions depending on the location in code (application or test environment)
		ExcpMissFunc	Exception in a generated stub
		ExcpBasicFunc	Exception in a support function of a stub
		ExcpDataProcess	Exception in data range monitoring function
		ExcpNULLInj	Exception after injection of a NULL pointer
		StdExcpC++	Standard exception from C++
		TermExcpC++	Termination exception from C++
		InvalidAddr	Access of an invalid address, general message if source cannot be exactly determined
		AddrIsReadOnly	Address is not writable
		AddrIsNotRW	Address is not readable and not writable
		AddrIsNULL	Address is NULL, e.g. passed to index checking
		NULLptrDeref	Dereference of a NULL pointer
		Uexit	Unexpected termination of a test, condition could not covered by any of the implemented exception handlers, probably due to an invalid address
File Access Error		FileHandleErr	File handling error (open, close, file access, not opened)
Non-terminating Loop		FileTooBig	Log-file too large, possibly an indication of an infinite loop condition
Possible Recursion		RecursExcp	Exception during exception handling
		Recursion	Stack overflow possibly due to recursion
Resource Leak		Resource leak	File not closed, malloc-memory not freed
Arithmetic Overflow	W	FpNan	Contents of floating point data is not a number
		FpInf	Contents of floating point data represents infinite
		intOverflow	Integer overflow occurred
Concurrency Issues		tbd	The support will be given soon.
Invariant Condition		WasAlwaysTrue	The condition was always true, possibly invariant condition
		WasAlwaysFalse	The condition was always false, possibly invariant condition
Timeout during Execution		TimeoutIntMonitor	The test run was terminated due to reaching the time limit, possibly a deadlock or the system hangs
Unreachable Code		WasNotReachedBlk	The block was never reached
		WasNotReachedCnd	The condition was never reached

BSSE System and Software Engineering

Std. Defect Types vs. Tools(1/2)

SCISYS

Id	Standard Defect Type	Astree			BugFinder			CodeProver			DCRTT			QAC		
		supp	330	450	supp	330	450	supp	330	450	supp	330	450	supp	330	450
1	Array Index Out-of-Bounds		x	x		x	x		x	x	x	x	x			
2	Assert failure		x	x		x	x		x	x	x	x	x			
3	Dangling Pointer					x										
4	Dereference of Invalid or NULL Pointer		x	x		x	x		x	x	x	x	x		x	x
5	File Access Error										x					
6	Format String Mismatch					x										
7	Invalid arithmetic operation		x	x											x	x
8	Invalid function pointer		x	x					x	x						
9	Invalid Return Statement															
10	Loss of Precision					x	x								x	x
11	Macro Use with Unintended Consequences															
12	Non-terminating Loop		x	x					x	x	x					
13	Possible Invalid Use of Function		x	x		x	x		x	x						
14	Possible Recursion		x	x							x					
15	Resource Leak										x					
16	Undefined Result								x	x					x	x
17	Uninitialized Variable		x	x		x	x		x	x					x	x
18	Unintended Use of Implicit Member Function															
19	Arithmetic Operation on NULL Pointer		x	x												
20	Arithmetic Overflow		x	x		x	x		x	x	o				x	x
21	Cast to pointer of incompatible types		x	x											x	x
22	Comparison of floating-point values														x	
23	Concurrency Issues		x	x		x	x		x	x						
24	Conflicting Declarations															
25	Incomplete List of Cases for enum-Type without default		x													
26	Intended Change of Invariant Data		x			x	x									

No contribution for late version

BSSE System and Software Engineering

Std. Defect Types vs. Tools(2/2)

SCISYS

Id	Standard Defect Type	Astree			BugFinder			CodeProver			DCRTT			QAC		
		supp	330	450	supp	330	450	supp	330	450	supp	330	450	supp	330	450
27	Invalid pointer operation		x	x												
28	Invariant Condition					x	x				x				x	x
29	Invariant Expression		x	x											x	x
30	Loss of Update					x										
31	Memory Size Mismatch					x	x									
32	Name overloading					x									x	x
33	Parameter Type Mismatch in Function Call		x	x												
34	Possible invalid arithmetic operation		x	x		x										
35	Possible invalid pointer operation															
36	Possible Loss of precision		x	x												
37	Possible misuse of signed integer					x	x									
38	Tainted Data															
39	Timeout during Execution										x	x	x			
40	Unnecessary Loop Construct														x	x
41	Unnecessary Operation														x	x
42	Unreachable Code		x	x		x	x		x	x	x				x	x
43	Unreliable arithmetic cast		x	x											x	x
44	Unreliable pointer cast					x										
45	Unused Result		x	x		x	x									
46	Change of Data expected, but missing															
47	Incomplete List of Cases for enum-Type with default															
48	Inconsistent Overloading															
49	Multiple return paths															
50	Security Issue															
51	Unintended Change of Data															
52	DefectTypeIgnore															

○ *No contribution for late version*

Coincidences vs. Criticality

#Coincidences of Tools	Counts / Early Version				
	Critical	Warning	Uncritical	Ignored	All
1	7510	20259	0	9957	37726
2	1810	1565	0	48	3423
3	257	17	0	0	274
4	15	4	0	0	19
5	0	0	0	0	0
	9592	21845	0	10005	41442

#Coincidences of Tools	Counts / Late Version				
	Critical	Warning	Uncritical	Ignored	All
1	7874	23104	0	9775	40753
2	1858	1158	0	4	3020
3	268	8	0	0	276
4	6	0	0	0	6
5	0	0	0	0	0
	10006	24270	0	9779	44055

Figures are related to consolidated set < sum of all reports

Coincidences vs. Tools (late)

Cnt	%	Tool Combinations / Late Version
2390	5.43	Tool1 Tool3
251	0.57	Tool1 Tool3 DCRTT
113	0.26	Tool1 DCRTT
141	0.32	Tool1 Tool2
49	0.11	Tool1 Tool5
5	0.01	Tool1 Tool2 Tool3 DCRTT
7	0.02	Tool1 Tool2 Tool3
1	0.00	Tool1 DCRTT Tool5
9	0.02	Tool1 Tool3 Tool5
1	0.00	Tool1 Tool2 Tool3 Tool5
2	0.00	Tool1 Tool2 Tool5
277	0.63	Tool2 Tool3
6	0.01	Tool2 Tool3 Tool5
1	0.00	Tool2 Tool5
2	0.00	Tool3 Tool5
38	0.09	Tool3 DCRTT
871	1.98	DCRTT
9	0.02	DCRTT Tool5
44055	100.00	Total

Figures are related to sum of consolidated reports 44055

D = DCRTT

Figures are related to consolidated set < sum of all reports

Coincidences vs. Criticality and Tools / *late* / *multi-tasking*

BSSE System and Software Engineering

SCISYS

All			Critical			Warning		
#Coinc.	Tool Comb.	%	#Coinc.	Tool Comb.	%	#Coinc.	Tool Comb.	%
369	T1 T2	0,69	7	T1 T2	0,01	354	T1 T2	0,66
22	T1 T2 T3	0,04	22	T1 T2 T3	0,04			
25	T1 T2 T3 D	0,05	25	T1 T2 T3 D	0,05			
4	T1 T2 T3 T5	0,01	4	T1 T2 T3 T5	0,01			
10	T1 T2 T5	0,02				10	T1 T2 T5	0,02
6929	T1 T3	12,99	4507	T1 T3	8,45	2422	T1 T3	4,54
1099	T1 T3 D	2,06	1099	T1 T3 D	2,06			
39	T1 T3 T5	0,07	39	T1 T3 T5	0,07			
302	T1 D	0,57	302	T1 D	0,57			
7	T1 D T5	0,01	7	T1 D T5	0,01			
103	T1 T5	0,19	15	T1 T5	0,03	88	T1 T5	0,17
1021	T1 T2	1,91	2	T1 T2	0,00	1019	T1 T2	1,91
22	T1 T2 T5	0,04				22	T1 T2 T5	0,04
3	T1 T5	0,01				3	T1 T5	0,01
80	T3 D	0,15	80	T3 D	0,15			
6	T3 T5	0,01				6	T3 T5	0,01
1097	D	2,06	1074	D	2,01	23	D	0,04
20	D T5	0,04	20	D T5	0,04			
53327	**All**	**100,00**	**14790**	**All**	**27,73**	**28648**	**All**	**53,72**

Figures are related to sum of all reports 53227

D = DCRTT

Report Summary

- **Configuration definition**
  - defined in contact with tool supplier when many options supported
  - iterations, if required
  - fairness: optimise results for each tool
- **Configuration update**
  - where possible, options were activated to get reports for defect types missing so far
  - then number of reports significantly increased, other defect types than intended were activated in addition
- **Results from manual evaluation**
  - DCRTT without context always a TP (precision 100%)
  - static analysers without context: precision < 100% due to (over-)approximation
- **Report filtering for TP detection**
  - guided by DCRTT filtering mechanisms and reduced number of reports (entry-point)
  - first detected with DCRTT, then checked for occurrence in reports of the other tools

Report Summary

Tool	Early Version				Late Version			
	entry-point-function			unit test	entry-point-function			unit test
	determ.	non-det.	multi-task.		determ.	non-det.	multi-task.	
DCRTT	31	31	n/a	1590	16	21	n/a	1638
Other Tools	One tool supplier denied publishing of data, therefore no figures are published with reference to a tool **range 800 – 29000** ***(entries in std. csv-file)*** ***(entries extracted from tool-specific report files)***							

more functions

- **Entry-point function DCRTT**
  - number of reports and false positives significantly decreased for entry-point function due to consideration of context
  - TP could easily identified, but low coverage (~20%) due to missing stimulation
    - future goal: injection of telecommands and stimulation of external interfaces
- **Unit testing DCRTT**
  - filtering mechanisms provided to prioritize reports
- **Other tools**
  - no filtering mechanisms

Evaluation Results for 6 TP

Tool	Version	Execution Mode	#Reports	TP		FP				
						with ctxt				w/o ctxt
				with ctxt	w/o ctxt	**tot**	data	task	stub	
DCRTT	330	deterministic	31	4	31	**27**	20	3	4	0
		non-deter.								
	450	deterministic	16	1	16	**15**	8	3	4	0
		non-deter.	21		21	**20**	13			

source of FP
non-representative analysis environment

4 + 1 from DCRTT entry-point execution
+ 1 from DCRTT unit testing

Vers.	Function	Location	Id	Reported by		
				DCRTT		Other Tools
				Entry-point-version	Module Testing	
early	*func1*	*X*	1	*determ. + non-determ.*	x	*2x*
		X+2	2			
		X+5	3			
	func2	*Y*	4			*2x*
late	*func3*	*Z*	5	*determ. + non-determ.*	none	*1x*
	func4	*U*	6	none	x	*2x*

The two issues for the late version – highlighted in the analysis – have no impact on the current operational concept and mission performance.

Evaluation Results for 6 TP

Tool	Version	Execution Mode	#Reports	TP		FP				
						with ctxt				w/o ctxt
				with ctxt	w/o ctxt	**tot**	data	task	stub	
DCRTT	330	deterministic	31	4	31	**27**	20	3	4	0
		non-deter.								
	450	deterministic	16	1	16	**15**	8	3	4	0
		non-deter.	21		21	**20**	13			

4 + 1 from DCRTT entry-point execution
+ 1 from DCRTT unit testing

Vers.	Function	Location	Id	Reported by		
				DCRTT		Other Tools
				Entry-point-version	Module Testing	
early	*func1*	*X*	1	*determ. + non-determ.*	x	*2x*
		X+2	2			
		X+5	3			
	func2	*Y*	4			*2x*
late	*func3*	*Z*	5	*determ. + non-determ.*	none	*2x*
	func4	*U*	6	none	x	*2x*

COPY for snapshor

Reporting Aspects: DCRTT

- reports only generated when a real anomaly is detected
- FP only may occur due to
  - missing context at the function interface
  - Porting inaccuracies
- provision of information on current context for defect activation
- priorisation by content-sensitive filtering for out-of-range reports

BSSE System and Software Engineering

Reporting Aspects: Abstract Interpretation

SCISYS

- **complexity may require approximation of the context**
- **loss of context implies reporting of false positives**
- **mapping of vectors (use of arrays) big challenge for context representation**
- **loss of context may increase coverage due to consideration of more combinations than for real context**
  - ⇨ but increased number of false positives due to invalid combinations
  - ⇨ loss of context ⇔ missing context, not only at the function interface as for DCRTT

    but also inside a function and following call tree
- **this may explain the number of reports and false positives**

Evaluation Issues

- **Static Analyser Reports**
  - all reports have same prior probability of being FP or TP
  - in presence of many reports (thousands and much more) ⇨ huge manual effort
  - sampling of reports ⇨ risk of missing TP (factual conversion to FN)
  - sound approach ends up as potentially unsound for the whole evaluation process
  - TP information from DCRTT was used to check reports of static analysers
  - in case of static analysers other TP found by evaluation of a subset
- **Required:**
  - information on context of the defect needed
  - indicator(s) highlighting potential defects
  - tool with high precision
- **TP evaluation**
  - in a reasonable number of cases full context crossed task boundaries
  - difficult to assess: is it a TP or not? Decided in favour of the tool as TP
  - difficult to obtain a fair comparison
  - about 60 reports evaluated manually for each version, up to 3 hours per report

Indicator Examples (1/2)

DCRTT Report for Out-of-Bounds 1

OutOfRangeHigh; testId=660, func=03271, block=1 arrId=16596 idx=1 Value=7, upLim=7 at <location>

idx=1 ⇨ first index of array with id 16596 is out-of-bounds

DCRTT: indicator for a potential TP

value=7 and upLim=7

IS filtered by DCRTT

Indicator Examples (2/2)

DCRTT Report for Out-of-Bounds 2

OutOfRangeHigh;testId=8, func=00589, block=8 idx=1 Value=4128, upLim=4096
stmt=memcpy_BSSE at <location>

idx=1 ⇨ dest-para of memcpy with size 4096 while 4128 bytes shall be copied

DCRTT: one report only

Static analyser reported about 600 possible combinations with invalid / impossible combinations of source, destination and size

Context Considerations

- **Defect reporting and DCRTT**
  - evaluation without context (robustness testing, security issues) always a TP
  - evaluation with context: issues on knowledge of the context

    issue can be solved by provision of semantic information
  - every report highlights an issue which possibly may be invalidated only by non-representative context generated in the test environment

- **DCRTT entry-point approach**
  - most of defect detection mechanisms available just as for unit testing
  - every report is a TP if the context is representative for the operational conditions
  - much less FP compared to unit testing with insufficient context information
  - less manual effort for evaluation

Lessons Learned

Sources of False Positives

- **Mismatch of Verification Criteria and Programming Style**
  - if verification criteria are not considered during development ⇨ high number of FP
  - verification tool(s) should be considered continuously over the development cycle
- **Non-representativity of the analysis environment**
  - just exposing the source code to the analysis may not be sufficient
  - e.g. stubbing, scheduling scheme, dynamic changes of object structure (telecmd.)
- **Non-representativity of the analysis method**
  - context vs. robustness trade-off and approach of chosen tool
  - provision of required context information by tool and user
- **Approximation of the context**
  - exact representation of the context vs. memory consumption
  - context information may be lost due to approximation
  - benefits of using context information may be lost ⇨ increased number of FP
- **Missing or non-visible checks**
  - checks not present to ensure valid conditions
  - checks present but not visible for the verification tool, e.g. task boundaries or ground checks

Context Approximation Example 1/2: interval approx.

```
typedef enum {
  lit0=0,lit1=1,lit2=2,
  litInvalid=255
} TySet;

TySet map2set(uint8 para){
  if (para==0) return lit0;
  else if (para==1) return lit1;
  else if (para==2) return lit2;
  else return litInvalid;
}

int myArr[lit2+1];

void myFunc(uint8 para) {
  idx=map2set(para);

  if (idx != litInvalid)
       myArr[idx]=0;
  return;
}
```

```
setEnum  =0,1,2,255 exact

setReturn=0,1,2,255 exact
setApprox=[0,255]    interval approx.

setApprox =[0,255]

setApprox2=[0,254], 255 removed
idx may be > 2: FP will be reported
```

Context Approximation Example 2/2: min/max

```
int myMapping[6]=
{1,3,5,54,7,78};

char mySrc [100];
char myDest[  5];

void myFunc(int idx) {
  if (idx>0 && idx<6) {
    memcpy(dest,
           src,
           myMapping[idx]);
  return;
}

int main(int argc, char* argv)
{
  myFunc(2);
  return 0;
}
```

array contents is approximated / squashing
by min/max: [1,78]

min/max are considered here:**[1,78]**

myFunc is called with
idx=2 ⇨ size=5 which is **valid**

but taking the maximum 78 the following report is issued:
78 out-of-bounds [0,5]

Coding Style vs. False Positives

⇩ Missing context

- highly desirable: support of constraints and correlations
- already practice in other domains like automotive
- valid for static and dynamic analysis tools

⇩ More precise code

- use of *const* whenever appropriate (global data)
- avoids stimulation / overriding of meaningful data

⇩ Context-independent units

- no external context, full set of checks in unit
- conflict: checks to be repeated in every task?

Conclusions and Outlook

Recommendations

- Ensure a representative context to the degree possible
- Choose the right tool approach for the envisaged verification goal

  robustness testing vs. pure unit testing, context-sensitive or not

  provide as much context-information as possible
- Consider the feedback from the verification tool(s) as early as possible during coding
- Fix the defects according to the tool feedback
- Discuss a trade-off on protection against invalid data

  check or do not check?

  Checking will reduce the amount of false positives

Conclusions on Evaluation

- **Evaluation Strategy**
  - a large amount of reports requires selection of a subset representative context for functions
  - filtering algorithms should be supported by a tool
  - filtering should provide report sets having a higher chance to evaluate as FP
- **Soundness**
  - soundness may be lost if evaluation is limited to a subset
  - soundness may be lost if report file does not include all reports on TP
- **Coincidences**
  - reports on the same defect and location do not necessarily suggest it is a TP
  - a single report at presence of more than one tool does not necessarily imply it is a FP
  - analysis may be limited to the intersection of report sets from sound tools
- **Context**
  - approximation of the context may cause more FP than missing context at the interface
- **Comparison early vs. late version**
  - in most cases more reports for early version, no indication for defect hiding

Evaluation Process

- **Manually evaluated reports: ~60 per version, in total ~120**
  - assessment for {tool / state criterion} x {with / w/o context} = 4 combinations
  - TP considered for state criterion / with context: most representative
  - evaluation effort rather high, up to 4 hours
  - originally intended 20 per tool and version

- **Selection criteria**
  - TP directly pointed to by DCRTT
  - TP identified during analysis of a TP
  - randomly selected reports from the critical set

- **Assessment for remaining TP**
  - assessment of TP on relevance (filtering: due to stubs, non-visible pre-checks)
  - relevant TP forwarded to developer team

Detection of Relevant TP

Version	Id	Ref[1]	Detected by	Detection Method	Confirmed	Reported by other Tools
early	1	1,2,3	DCRTT	directly by index reporting		2
	2	4		*in filtered file*		2
	3	*added*[2]		sampling		2
late	1	5		directly by index reporting	yes	2
	2	6		*in filtered file*	yes	2
	3	*added*		sampling / comparison early/late	no	2
	4	*added*		sampling	no	2

late/1: the report from a (sound) tool was not in the (standard) tool report, difficult to find

The smaller the report set, the higher the probability to detect TP candidates by sampling

(1)Reference to previous slide „Evaluation Results for 6 TP

(2) Candidate TP added from sampling

Detection of Relevant TP

Version	Id	Ref	Detected by	Detection Method	Confirmed	Reported by other Tools
early	1	1,2,3	DCRTT	directly by index reporting		2
	2	4		*in filtered file*		2
	3	*added*		sampling		2
late	1	5		directly by index reporting	yes	2
	2	6		*in filtered file*	yes	2
	3	*added*		sampling / comparison early/late	no	2
	4	*added*		sampling	no	2

late/1: the report from a (sound) tool was not in the (standard) tool report, difficult to find

**The smaller the report set,
the higher the probability to detect
TP candidates by sampling**

COPY for SNAPSHOT

Final Evaluation Results (early)

Report Assessment	Tool w/o ctxt	State w/o ctxt	Tool with ctxt	State with ctxt
TP	39	37	10	**8**
assert	8	8	3	**2**
out-of-bounds	21	21	5	**5**
dereference	8	8	0	**0**
uninitialised	1	0	1	**0**
undefined result	1	0	1	**0**
FP	3	5	32	**34**
Total	42	42	42	42

Consolidated reports 42 < 58 manually evaluated reports

Relevance	Assessment	Number of TP Reports
should be fixed	Index out-of-bounds	2
not relevant	Assertion failure due to stub	1
not relevant	Supposing that telecommand contents is checked on-ground or in another task, check not visible	4
not relevant	Assertion always fails due to coding	1

Final Evaluation Results (late)

Report Assessment	Tool w/o ctxt	State w/o ctxt	Tool with ctxt	State with ctxt
TP	48	46	11	**9**
assert	7	7	1	**1**
out-of-bounds	34	33	9	**8**
dereference	6	6	0	**0**
uninitialised	1	0	1	**0**
FP	8	10	45	**47**
Total	56	56	56	56

Consolidated reports 56 < 57 manually evaluated reports

Relevance	Assessment	Number of TP Reports	
		In 450	Overlap with 330
should be fixed	one-of-index fault (1x), invalid index (1x)	2	0
not relevant	hidden check	3	0
not relevant	Assertion failure due to stub	1	1
not relevant	Supposing that telecommand contents is checked on-ground or in another task, check not visible	3	3

DCRTT Potential

- **Defect detection**
  - evaluation without context (robustness testing, security issues) always a TP
  - evaluation with context: issues on knowledge of the context

    issue can be solved by provision of semantic information
  - every report highlights an issue which possibly may be invalidated only by non-representative context generated in the test environment

- **DCRTT entry-point approach**
  - most of defect detection mechanisms available as for unit testing
  - every report is a TP if the context is representative for the operational conditions
  - much less FP if the context is not representative
  - less manual effort for evaluation

Outlook DCRTT

- **Entry-Point Function**
  - the significant reduction of FP is impressing due to representative context for functions
  - TC injector based on inputs in EDS / XML format will be considered
  - stimulation moves from module level to system level
  - still a challenge: provision of schedule information to reduce related FP
- **Untyped byte streams**
  - genetic algorithms on unit level
  - first experience quite promising
- **Coverage**
  - future increased use of constrained-based test data generation
  - extension of the support
- **Open Interface**
  - expecting use of the link

Proposed Test Strategy

- **Testing in representative context**
  - entry-point function / integrated system and full DCRTT instrumentation
    - telecommand injection and stimulation of external interfaces
    - representative scheduling
    - constrained-based testing and genetic algorithms for the subset of functions for missing coverage
  - full set of functions for unit testing complemented by constraints and correlations not found automatically
- **Robustness testing**
  - DCRTT and full set of functions with fault injection activated invalid data / no constraints and fault-injection-wrappers
  - entry-point function / integrated system with fault-injection wrappers

BSSE System and Software Engineering

SCISYS

Thank you for your attention !

Questions?

Back-up

Example VectorCast Test Script scalar parameters

```
-- Test Case Script
-- Environment        :
vc_test_hello_vcast_1_BSSE_main3_9
-- Function Under Test: hello_vcast_1 - BSSE_main3
9 of hello_vcast_1.c
-- Script Features
TEST.SCRIPT_FEATURE:C_DIRECT_ARRAY_INDEXING
TEST.SCRIPT_FEATURE:CPP_CLASS_OBJECT_REVISION
TEST.SCRIPT_FEATURE:MULTIPLE_UUT_SUPPORT
TEST.SCRIPT_FEATURE:STANDARD_SPACING_R2
TEST.SCRIPT_FEATURE:OVERLOADED_CONST_SUPPORT
-- End of header hello_vcast_1 - BSSE_main3 9 of
hello_vcast_1.c
--  vc_test_9.tst generated by
dcrtt_open_if_cnv_vc.c for tool vc on <date>
-- Test File: hello_vcast_1.c
TEST.UNIT:hello_vcast_1
TEST.SUBPROGRAM:BSSE_main3
-- mangled name BSSE_main3
-- List of relevant data of function BSSE_main3
-- #tot    para=  2
-- #func   para=  2
-- #glob   para=  0
-- #constr para=  0
-- Parameters
--  signed int  argc
--  char ** argv
-- Return
--  signed int _return_
```

```
TEST.NEW
TEST.NAME:(CL)BSSE_main3.001
-- derived from DCRTT test case     1
TEST.NOTES:
No requirements provided
TEST.END_NOTES:
TEST.FLOATING_POINT_TOLERANCE: 9.9999997473787516000e-006
-- malloc for global data
--    no malloc for globals required
-- malloc for parameters
TEST.VALUE:hello_vcast_1.BSSE_main3.argv: <<malloc 1>>
TEST.VALUE:hello_vcast_1.BSSE_main3.argc: -2147483648
TEST.VALUE:hello_vcast_1.BSSE_main3.argv: ""
TEST.VALUE:hello_vcast_1.BSSE_main3.return: -2147483648
TEST.EXPECTED_USER_CODE:hello_vcast_1.BSSE_main3.argc
{{ (signed long)<<hello_vcast_1.BSSE_main3.argc>> == ( (signed
long)-2147483648) }}
TEST.END_EXPECTED_USER_CODE:
TEST.EXPECTED_USER_CODE:hello_vcast_1.BSSE_main3.argv
{{ <<hello_vcast_1.BSSE_main3.argv>> ==  "" }}
TEST.END_EXPECTED_USER_CODE:
TEST.EXPECTED_USER_CODE:hello_vcast_1.BSSE_main3.return
{{ (signed long)<<hello_vcast_1.BSSE_main3.return>> == ( (signed
long)0) }}
TEST.END_EXPECTED_USER_CODE:
TEST.END
```

Function Call Coverage Report

Configuration Data

Date of Report Creation: 25 MAY 2018
Time of Report Creation: 11:51:35 PM

Function Call Coverage

Unit	Functions	Function Coverage	Function Calls	Uncovered Calls
hello_vcast_1.c'1	checkOORdown	Y		
	checkOORup	Y		
	checkPtrReturn	Y		
	checkMissingItems	N	0 / 5 (0%)	DCRTT_stdout_fprintf(4 2)
				DCRTT_stdout_fprintf(4 5)
				missFunc(4 6)
				DCRTT_stdout_fprintf(4 7)
				DCRTT_stdout_fprintf(4 14)
	checkMissingItemsAndParas	N	0 / 5 (0%)	DCRTT_stdout_fprintf(5 2)
				DCRTT_stdout_fprintf(5 5)
				missFunc(5 6)
				DCRTT_stdout_fprintf(5 7)
				DCRTT_stdout_fprintf(5 18)
	checkGlobDataInitNULL	Y		
	checkGlobDataNonInitNULL	Y		
	BSSE_main	N	0 / 1 (0%)	DCRTT_stdout_fprintf(8 1)
	BSSE_main2	Y	1 / 1 (100%)	
	BSSE_main3	Y	1 / 1 (100%)	
TOTALS		**7 / 10 (70%)**	**2 / 13 (15%)**	
hello_vcast_1.c'2	checkOORdown	Y		
	checkOORup	Y		
	checkPtrReturn	Y		
	checkMissingItems	N	0 / 5 (0%)	DCRTT_stdout_fprintf(4 2)
				DCRTT_stdout_fprintf(4 5)
				missFunc(4 6)
				DCRTT_stdout_fprintf(4 7)
				DCRTT_stdout_fprintf(4 14)
	checkMissingItemsAndParas	N	0 / 5 (0%)	DCRTT_stdout_fprintf(5 2)
				DCRTT_stdout_fprintf(5 5)
				missFunc(5 6)
				DCRTT_stdout_fprintf(5 7)
				DCRTT_stdout_fprintf(5 18)
	checkGlobDataInitNULL	Y		
	checkGlobDataNonInitNULL	Y		
	BSSE_main	N	0 / 1 (0%)	DCRTT_stdout_fprintf(8 1)
	BSSE_main2	Y	1 / 1 (100%)	
	BSSE_main3	Y	1 / 1 (100%)	
TOTALS		**7 / 10 (70%)**	**2 / 13 (15%)**	
GRAND TOTALS		**14 / 20 (70%)**	**4 / 26 (15%)**	

Environment Coverage Report

Configuration Data

Date of Report Creation: 25 MAY 2018
Time of Report Creation: 11:51:27 PM

Environment Coverage

File / Environment	Statements	Branches	Pairs	Function Calls
hello_vcast_1.c'1	**32 / 72 (44%)**	**43 / 102 (42%)**	**9 / 23 (39%)**	**2 / 13 (15%)**
doif_config / GNU_Native_6.3_C / vc_test_hello_vcast_1_host	32 / 72 (44%)	43 / 102 (42%)	9 / 23 (39%)	
hello_vcast_1.c'2	**32 / 72 (44%)**	**43 / 102 (42%)**	**9 / 23 (39%)**	**2 / 13 (15%)**
doif_config / GNU_Native_6.3_C / vc_test_hello_vcast_1_tgt	32 / 72 (44%)	43 / 102 (42%)	9 / 23 (39%)	

Metrics Report

Configuration Data

Date of Report Creation: 25 MAY 2018
Time of Report Creation: 11:51:19 PM

Metrics

Unit	Subprogram	Complexity	Function Calls	Statements	Branches	Pairs
hello_vcast_1.c'1	checkOORdown	2		4 / 4 (100%)	5 / 5 (100%)	1 / 1 (100%)
	checkOORup	2		4 / 4 (100%)	5 / 5 (100%)	1 / 1 (100%)
	checkPtrReturn	2		4 / 4 (100%)	5 / 5 (100%)	1 / 1 (100%)
	checkMissingItems	6	0 / 5 (0%)	0 / 15 (0%)	0 / 21 (0%)	0 / 5 (0%)
	checkMissingItemsAndParas	8	0 / 5 (0%)	0 / 19 (0%)	0 / 29 (0%)	0 / 7 (0%)
	checkGlobDataInitNULL	2		4 / 4 (100%)	5 / 5 (100%)	1 / 1 (100%)
	checkGlobDataNonInitNULL	2		4 / 4 (100%)	5 / 5 (100%)	1 / 1 (100%)
	BSSE_main	3	0 / 1 (0%)	0 / 6 (0%)	0 / 9 (0%)	0 / 2 (0%)
	BSSE_main2	3	1 / 1 (100%)	6 / 6 (100%)	9 / 9 (100%)	2 / 2 (100%)
	BSSE_main3	3	1 / 1 (100%)	6 / 6 (100%)	9 / 9 (100%)	2 / 2 (100%)
TOTALS	**10**	**33**	**2 / 13 (15%)**	**32 / 72 (44%)**	**43 / 102 (42%)**	**9 / 23 (39%)**
hello_vcast_1.c'2	checkOORdown	2		4 / 4 (100%)	5 / 5 (100%)	1 / 1 (100%)
	checkOORup	2		4 / 4 (100%)	5 / 5 (100%)	1 / 1 (100%)
	checkPtrReturn	2		4 / 4 (100%)	5 / 5 (100%)	1 / 1 (100%)
	checkMissingItems	6	0 / 5 (0%)	0 / 15 (0%)	0 / 21 (0%)	0 / 5 (0%)
	checkMissingItemsAndParas	8	0 / 5 (0%)	0 / 19 (0%)	0 / 29 (0%)	0 / 7 (0%)
	checkGlobDataInitNULL	2		4 / 4 (100%)	5 / 5 (100%)	1 / 1 (100%)
	checkGlobDataNonInitNULL	2		4 / 4 (100%)	5 / 5 (100%)	1 / 1 (100%)
	BSSE_main	3	0 / 1 (0%)	0 / 6 (0%)	0 / 9 (0%)	0 / 2 (0%)
	BSSE_main2	3	1 / 1 (100%)	6 / 6 (100%)	9 / 9 (100%)	2 / 2 (100%)
	BSSE_main3	3	1 / 1 (100%)	6 / 6 (100%)	9 / 9 (100%)	2 / 2 (100%)
TOTALS	**10**	**33**	**2 / 13 (15%)**	**32 / 72 (44%)**	**43 / 102 (42%)**	**9 / 23 (39%)**
GRAND TOTALS	**20**	**66**	**4 / 26 (15%)**	**64 / 144 (44%)**	**86 / 204 (42%)**	**18 / 46 (39%)**

Example Cantata Test Script scalar parameters 1/2

```
/* TCcnt=1 mode=rnd  by coverage only */
 void checksAgainstDCRTToutputRandom_M1_1(){
                        int DCRTTcheckCnt=0,DCRTTdiffCnt=0,DCRTTcheckVal=0;
                        DCRTTdiffCnt += DCRTTcheckVal;
 /* Test Parameter Check: arr ; funcPara */
        DCRTTcheckCnt++;
        DCRTTcheckVal=CHECK_S_INT_DCRTT((signed long)_arr_[0],(signed long)-2147483648);
        DCRTTdiffCnt+=DCRTTcheckVal;
        DCRTT_stdout_fprintf(getFormatStringAssertion(),1-
DCRTTcheckVal,DCRTT_DIFF_OUT_STR,DCRTTcheckCnt,"_arr_[0]",getAssertionString(DCRTTcheckVal),"INOUT FUNCPARA");
 /* Section end of parameter check: arr */
 /* Test Parameter Check: return ; return */
        DCRTTcheckCnt++;
        DCRTTcheckVal=CHECK_S_INT_DCRTT((signed long)_retDCRTT_,(signed long)0);
        DCRTTdiffCnt+=DCRTTcheckVal;
        DCRTT_stdout_fprintf(getFormatStringAssertion(),1-
DCRTTcheckVal,DCRTT_DIFF_OUT_STR,DCRTTcheckCnt,"_retDCRTT_",getAssertionString(DCRTTcheckVal),"RETURN");
 /* Section end of parameter check: return */
                        DCRTTdiffCnt=DCRTTcheckCnt-DCRTTdiffCnt;
                        DCRTT_stdout_fprintf(" %d differences of %d in output for test Random_M1_" "1" "
                                ASSERTION_TRACE_OUT_OUT_SUM\n",DCRTTdiffCnt,DCRTTcheckCnt);
                        fflush(fdCompare);
}
#endif /* DCRTT_NO_CHECKS_ON_RESULTS */
void compareParaValues(char *testCaseId){
FUNCBEGINcantpp
        if (fdCompare) DCRTT_FPRINTF(fdCompare,"\n/* ========================================= */\n");
        if (fdCompare) DCRTT_FPRINTF(fdCompare,"\n/* Identification of changes for test case %s */\n",testCaseId);
        sprintf(cantppTestLside,"_arr_");
        sprintf(cantppTestRside,"_arrInput_");
        cmpCnt        =0;
        actAssignStmts=0;
```

Example Cantata Test Script scalar parameters 2/2

```
 compareDCRTT_arr          (compEQ,&cmpCnt,prtCtrlEQ,fdCompare,1,&actAssignStmts,5000,cantppTestLside,cantppTestRside,"
",&_arrInput_,&_arr_);
        compareDCRTT_arr
(compEQ,&cmpCnt,prtCtrlEQ,fdCompare,1,&actAssignStmts,5000,cantppTestLside,cantppTestRside,"        ",&_arrInput_,&_arr_);
        if (fdCompare) DCRTT_FPRINTF(fdCompare," %d items out of %d are identical for arr regarding IN - OUT comparison of
PARA 0 ASSERTION_TRACE_IN_OUT_IDENT INOUT FUNCPARA \n",cmpCnt,actAssignStmts);cmpCnt        =0;
        actAssignStmts=0;
        compareDCRTT_arr
(compNE,&cmpCnt,prtCtrlNE,fdCompare,1,&actAssignStmts,5000,cantppTestLside,cantppTestRside,"        ",&_arrInput_,&_arr_);
        compareDCRTT_arr
(compNE,&cmpCnt,prtCtrlNE,fdCompare,1,&actAssignStmts,5000,cantppTestLside,cantppTestRside,"        ",&_arrInput_,&_arr_);
        if (fdCompare) DCRTT_FPRINTF(fdCompare," %d items out of %d are different for arr regarding IN - OUT comparison of
PARA 0 ASSERTION_TRACE_IN_OUT_DIFF INOUT FUNCPARA \n",cmpCnt,actAssignStmts);
        strcpy(cantppTestLside,"_retDCRTT_");
        sprintf(cantppTestRside,"_retDCRTT_Input");
        cmpCnt        =0;
        actAssignStmts=0;
        compareDCRTT__return_
(compEQ,&cmpCnt,prtCtrlEQ,fdCompare,1,&actAssignStmts,5000,cantppTestLside,cantppTestRside,"
",&_retDCRTT_Input,&_retDCRTT_);
        if (fdCompare) DCRTT_FPRINTF(fdCompare," %d items out of %d are identical for _retDCRTT_ regarding IN - OUT
comparison of RETURN ASSERTION_TRACE_IN_OUT_IDENT RETURN \n",cmpCnt,actAssignStmts);
        cmpCnt        =0;
        actAssignStmts=0;
        compareDCRTT__return_
(compNE,&cmpCnt,prtCtrlNE,fdCompare,1,&actAssignStmts,5000,cantppTestLside,cantppTestRside,"
",&_retDCRTT_Input,&_retDCRTT_);
        if (fdCompare) DCRTT_FPRINTF(fdCompare," %d items out of %d are different for _retDCRTT_ regarding IN - OUT
comparison of RETURN ASSERTION_TRACE_IN_OUT_DIFF RETURN \n",cmpCnt,actAssignStmts);
        if (fdCompare) DCRTT_FPRINTF(fdCompare,"\n/* ========================================== */\n\n");
        if (fdCompare) fflush(fdCompare);
FUNCENDcantp
}
```

Output Cantata for Test Script Project Information

Cantata Test Report

DCRTT_CANTPP_PROJECT_host

Generated 25th April 2018 21:26

Project Information

Test Script Info

Summary status	Failed
Number of Test Scripts	18
Passed Test Scripts	9
Failed Test Scripts	9
Total number of test cases	117
Test cases passed	53
Test cases failed	64
Checks passed	979
Checks failed	80
Checks warning	0

Filter Information

Filters Applied: None

Coverage Achieved

Coverage Type	Target State	Coverage Achieved	Fully Covered Functions	Uninstrumented Functions
Block	Enabled	90%	18/27	0
Statement	Enabled	85%	18/27	0
Decision	Enabled	92%	25/27	0

Output Cantata for Test Script / Test Results Overview

Test Results

Test Script: BSSE_main

Test Script Description: = Cantata Test Harness v7.2 = = (c) 2017 QA Systems GmbH = = Test Started: Wed Apr 25 20:45:00 2018 =
Test Script File: C:\Projekte\dcrtt\mytest\testsrcdoif_cantpp12_all_files\dcrtt\dcrtt_xr\d_hello_cantpp_1_c\func7\testFunc7_cantpp.c

Summary status	Failed
Number of test cases	1
Test cases passed	0
Test cases failed	1
Checks passed	26
Checks failed	2
Checks warning	0
Script directives	0
Script errors	0
Script warnings	0
Call sequence failures	0
User comments	3

Test Case: Random_M1_00000001 - nominal data, exception occurs, NO fault injected

Summary status	Failed
Checks passed	26
Checks failed	2
Checks warning	0
Script directives	0
Script errors	0
Script warnings	0
Call sequence failures	0
User comments	3

Output Cantata for Test Script / Result Check

User Comment

Source File: C:\Projekte\dcrtt\mytest\testsrcdoif_cantpp12_all_files\dcrtt\dcrtt_xr\d_hello_cantpp_1_c\func7\testFunc7_cantpp.c
Line Number: 423

+++ Message from DCRTT:

User Comment

Source File: C:\Projekte\dcrtt\mytest\testsrcdoif_cantpp12_all_files\dcrtt\dcrtt_xr\d_hello_cantpp_1_c\func7\testFunc7_cantpp.c
Line Number: 758

*** ERROR exception did not occur in 0x0000039c HIGH_OUT_OF_RANGE_ERROR hello_cantpp_1.c:6680 excMsg # function BSSE_main # : Random_M1_00000001

User Comment

Source File: C:\Projekte\dcrtt\mytest\testsrcdoif_cantpp12_all_files\dcrtt\dcrtt_xr\d_hello_cantpp_1_c\func7\testFunc7_cantpp.c
Line Number: 431

Check: exceptionCheck = 1 Failed

Source File: C:\Projekte\dcrtt\mytest\testsrcdoif_cantpp12_all_files\dcrtt\dcrtt_xr\d_hello_cantpp_1_c\func7\testFunc7_cantpp.c
Line Number: 774

Expected value: 1
Actual value: 0

Check: (signed long)(*_argc_) = (signed long)-2147483648 Passed

Source File: C:\Projekte\dcrtt\mytest\testsrcdoif_cantpp12_all_files\dcrtt\dcrtt_xr\d_hello_cantpp_1_c\func7\testFunc7_cppDCRTTresults.c
Line Number: 115

Expected value: -2147483648
Actual value: -2147483648

Check: (char*)_argv_[0] = "lxivmf{lurnmkdzwlqrrjrjqg" Passed

Source File: C:\Projekte\dcrtt\mytest\testsrcdoif_cantpp12_all_files\dcrtt\dcrtt_xr\d_hello_cantpp_1_c\func7\testFunc7_cppDCRTTresults.c
Line Number: 122

Expected value: lxivmf{lurnmkdzwlqrrjrjqg
Actual value: lxivmf{lurnmkdzwlqrrjrjqg

BSSE System and Software Engineering

Output Cantata for Test Script / Result Summary

SCISYS

Cantata Test Results Summary

Project: *DCRTT_CANTPP_PROJECT_host* — **Overall Result: Fail**

Summary Information

Hostname	herakles	**Cantata version**	v7.2
Summary generated	Apr 25, 2018, 9:26 PM	**Time elapsed during test run**	Unknown

Build Summary

Total tests	55
Compile attempted (files)	0
Link attempted (tests)	9
Execute attempted (tests)	0

Results Summary

PASSED	0
FAILED	9
Total checks	1059
Total checks failed	80
Total script errors/call failures	0

Coverage Summary

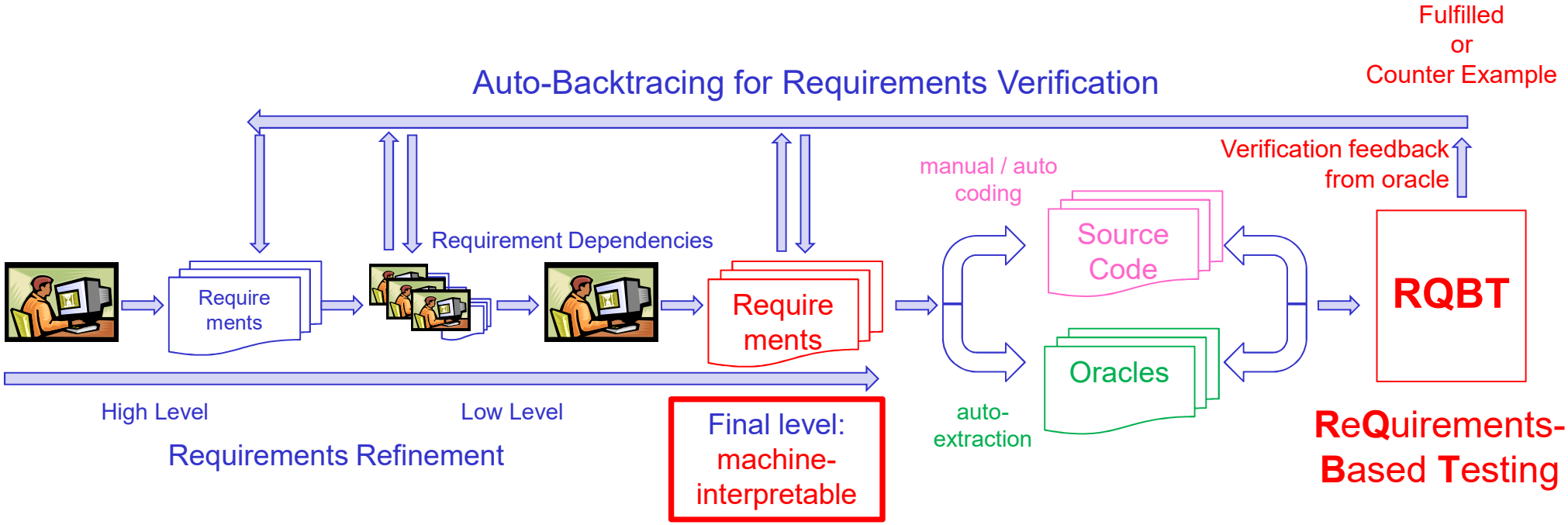
Entry point (E)	-
Statement (S)	85%
Decision (D)	92%
Call-return (C)	-
MC/DC - masking (M)	-
MC/DC - unique cause (U)	-

Failures

Note: The external links contained in this results summary will only work if this file is opened from the location of generation, and all the required log files are available.

Executable	Compiled	Linked	Executed	Script errors/call failures	Checks failed	Coverage (%)					
						E	S	D	C	M	U
testFunc0_cantpp	-	-	-	-	-	-	-	-	-	-	-
testFunc10_cantpp	-	-	-	-	-	-	-	-	-	-	-
testFunc12_cantpp	-	-	-	-	-	-	-	-	-	-	-
testFunc13_cantpp	-	-	-	-	-	-	-	-	-	-	-
testFunc3_cantpp	-	-	-	-	-	-	-	-	-	-	-
testFunc4_cantpp	-	-	-	-	-	-	-	-	-	-	-
testFunc7_cantpp	-	-	-	-	-	-	-	-	-	-	-
testFunc8_cantpp	-	-	-	-	-	-	-	-	-	-	-
testFunc9_cantpp	-	-	-	-	-	-	-	-	-	-	-

Logic Flow for RQBT



Only
required on this final level!

Challenge:
Notation for machine-readable code

Impact by Non-representative Analysis Environment

- **function stubbing**
  - stubs may / will not provide representative output
  - e.g. possible conflict between coverage and random output and assertions
- **scheduling**
  - calling task entries in a non-representative sequence will produce FP
  - the entry-point function must consider the scheduling concept
- **dynamic change of context**
  - the format of telecommands can be described formally
  - but the overall structure is dynamically defined at run-time
  - the tool gets an untyped byte stream
  - challenge for static analysers, source of many FP

Duplicates for Snapshots

Std. Defect Types vs. Tools(1/2)

SCISYS

Id	Standard Defect Type	Tool1			Tool2			Tool3			DCRTT			Tool5		
		supp	early	late	supp	early	late	supp	early	late	supp	early	late	supp	early	late
1	Array Index Out-of-Bounds		x	x		x	x		x	x	x	x	x			
2	Assert failure		x	x		x	x		x	x	x	x	x			
3	Dangling Pointer					x										
4	Dereference of Invalid or NULL Pointer		x	x		x	x		x	x	x	x	x		x	x
5	File Access Error										x					
6	Format String Mismatch					x										
7	Invalid arithmetic operation		x	x											x	x
8	Invalid function pointer		x	x					x	x						
9	Invalid Return Statement															
10	Loss of Precision					x	x								x	x
11	Macro Use with Unintended Consequences															
12	Non-terminating Loop		x	x					x	x	x					
13	Possible Invalid Use of Function		x	x		x	x		x	x						
14	Possible Recursion		x	x							x					
15	Resource Leak										x					
16	Undefined Result								x	x					x	x
17	Uninitialized Variable		x	x		x	x		x	x					x	x
18	Unintended Use of Implicit Member Function															
19	Arithmetic Operation on NULL Pointer		x	x												
20	Arithmetic Overflow		x	x		x	x		x	x	o				x	x
21	Cast to pointer of incompatible types		x	x											x	x
22	Comparison of floating-point values														x	
23	Concurrency Issues		x	x		x	x		x	x						
24	Conflicting Declarations															
25	Incomplete List of Cases for enum-Type without default		x													
26	Intended Change of Invariant Data		x			x	x									

○ *No contribution for late version*

COPY ANONYM

BSSE System and Software Engineering

Std. Defect Types vs. Tools(2/2)

SCISYS

Id	Standard Defect Type	Tool1			Tool2			Tool3			DCRTT			Tool5		
		supp	early	late	supp	early	late	supp	early	late	supp	early	late	supp	early	late
27	Invalid pointer operation		x	x												
28	Invariant Condition					x	x				x				x	x
29	Invariant Expression		x	x			○								x	x
30	Loss of Update					x	○									
31	Memory Size Mismatch					x	x									
32	Name overloading					x	○								x	x
33	Parameter Type Mismatch in Function Call		x	x			○									
34	Possible invalid arithmetic operation		x	x		x	○									
35	Possible invalid pointer operation															
36	Possible Loss of precision		x	x												
37	Possible misuse of signed integer					x	x									
38	Tainted Data															
39	Timeout during Execution										x	x	x			
40	Unnecessary Loop Construct														x	x
41	Unnecessary Operation														x	x
42	Unreachable Code		x	x		x	x		x	x	x				x	x
43	Unreliable arithmetic cast		x	x			○								x	x
44	Unreliable pointer cast					x	○									
45	Unused Result		x	x		x	x									
46	Change of Data expected, but missing															
47	Incomplete List of Cases for enum-Type with default															
48	Inconsistent Overloading															
49	Multiple return paths															
50	Security Issue															
51	Unintended Change of Data															
52	DefectTypeIgnore															

○ *No contribution for late version*

COPY ANONYM