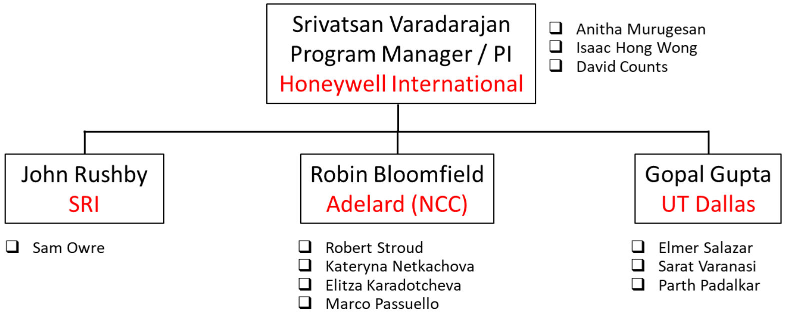
DARPA AUTOMATIC RAPID CERTIFICATION OF SOFTWARE (ARCOS)

Consistent Logical Automated Reasoning for Integrated System Software Assurance (CLARISSA)

ARCOS Technical Area 3



Srivatsan Varadarajan
Program Manager / PI
Honeywell International

- ❑ Anitha Murugesan
- ❑ Isaac Hong Wong
- ❑ David Counts

John Rushby
SRI

- ❑ Sam Owre

Robin Bloomfield
Adelard (NCC)

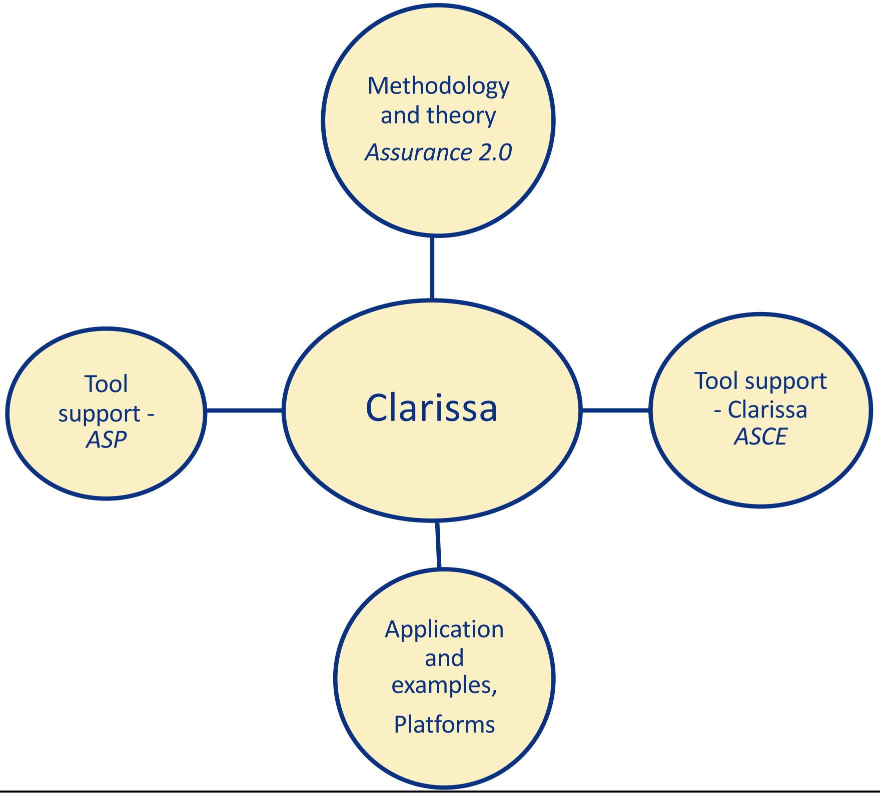
- ❑ Robert Stroud
- ❑ Kateryna Netkachova
- ❑ Elitza Karadotcheva
- ❑ Marco Passuello

Gopal Gupta
UT Dallas

- ❑ Elmer Salazar
- ❑ Sarat Varanasi
- ❑ Parth Padalkar

Disclaimers: This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA). The views, opinions and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

Distribution Statement "A“: Approved for Public Release, Distribution Unlimited.
If you have any questions, please contact the DARPA Public Release Center (PRC)



Clarissa

Methodology and theory *Assurance 2.0*

Tool support - *ASP*

Tool support - Clarissa *ASCE*

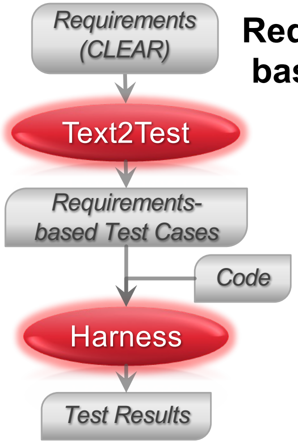
Application and examples,

Platforms

Distribution Statement "A": Approved for Public Release, Distribution Unlimited.
If you have any questions, please contact the DARPA Public Release Center (PRC)

THEORIES

A PRINCIPLED APPROACH TO DEFINING REUSABLE ASSURANCE FRAGMENTS



Requirements (CLEAR)

Text2Test

Requirements-based Test Cases

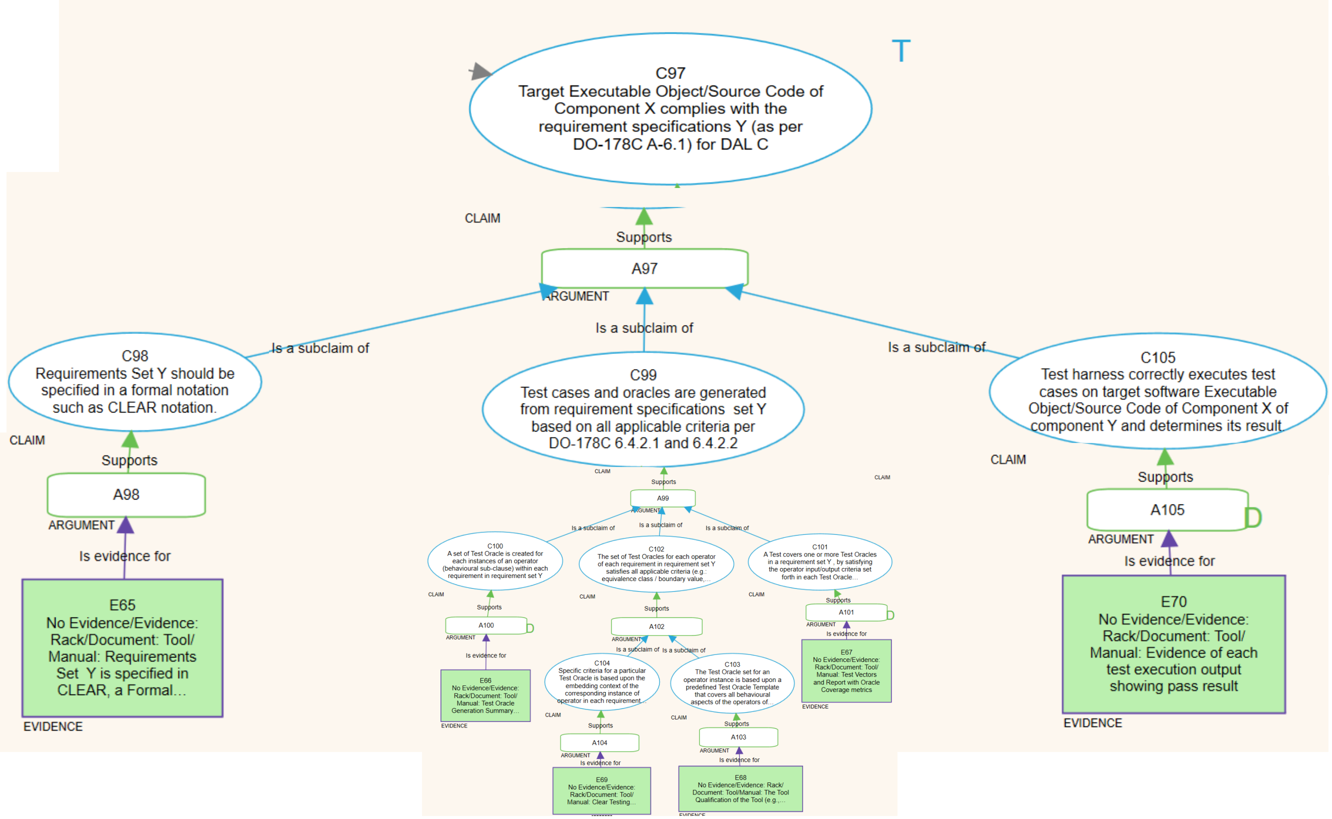
Code

Harness

Test Results

Requirements-based testing theory

Implementation conforms to requirements



T

C97
Target Executable Object/Source Code of Component X complies with the requirement specifications Y (as per DO-178C A-6.1) for DAL C

CLAIM

Supports

A97

ARGUMENT

Is a subclaim of

Is a subclaim of

Is a subclaim of

C98
Requirements Set Y should be specified in a formal notation such as CLEAR notation.

CLAIM

Supports

A98

ARGUMENT

Is evidence for

E65
No Evidence/Evidence: Rack/Document: Tool/ Manual: Requirements Set Y is specified in CLEAR, a Formal…

EVIDENCE

C99
Test cases and oracles are generated from requirement specifications set Y based on all applicable criteria per DO-178C 6.4.2.1 and 6.4.2.2

CLAIM

CLAIM

Supports

A99

ARGUMENT

Is a subclaim of

Is a subclaim of

Is a subclaim of

C100
A set of Test Oracle is created for each instances of an operator (behavioural sub-clause) within each requirement in requirement set Y

CLAIM

Supports

A100

ARGUMENT

Is evidence for

E66
No Evidence/Evidence: Rack/Document: Tool/ Manual: Test Oracle Generation Summary…

EVIDENCE

C102
The set of Test Oracles for each operator of each requirement in requirement set Y satisfies all applicable criteria (e.g.: equivalence class / boundary value,…

CLAIM

Supports

A102

ARGUMENT

Is a subclaim of

Is a subclaim of

C104
Specific criteria for a particular Test Oracle is based upon the embedding context of the corresponding instance of operator in each requirement…

CLAIM

Supports

A104

ARGUMENT

Is evidence for

E69
No Evidence/Evidence: Rack/Document: Tool/ Manual: Clear Testing…

C103
The Test Oracle set for an operator instance is based upon a predefined Test Oracle Template that covers all behavioural aspects of the operators of…

CLAIM

Supports

A103

ARGUMENT

Is evidence for

E68
No Evidence/Evidence: Rack/ Document: Tool/Manual: The Tool Qualification of the Tool (e.g.,…

EVIDENCE

C101
A Test covers one or more Test Oracles in a requirement set Y , by satisfying the operator input/output criteria set forth in each Test Oracle…

CLAIM

Supports

A101

ARGUMENT

Is evidence for

E67
No Evidence/Evidence: Rack/Document: Tool/ Manual: Test Vectors and Report with Oracle Coverage metrics

EVIDENCE

C105
Test harness correctly executes test cases on target software Executable Object/Source Code of Component X of component Y and determines its result

CLAIM

Supports

A105

ARGUMENT

Is evidence for

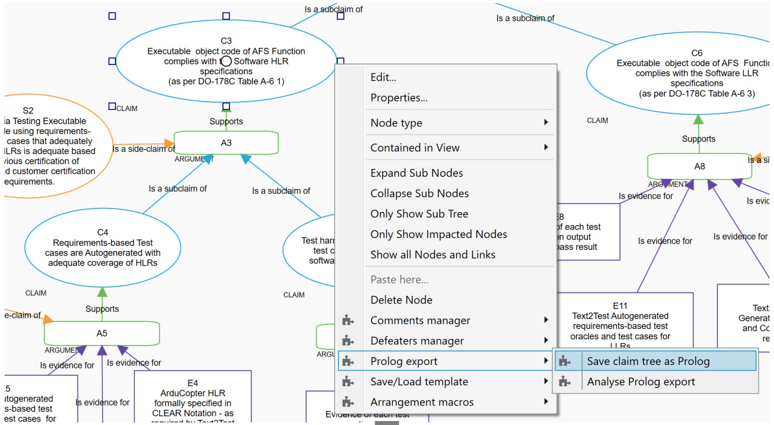
E70
No Evidence/Evidence: Rack/Document: Tool/ Manual: Evidence of each test execution output showing pass result

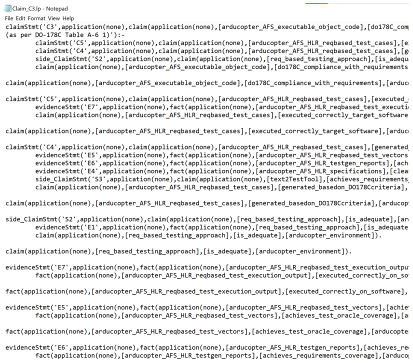
EVIDENCE

ASCE AUTOMATIC PROLOG EXPORT FOR S(CASP) ENGINE

- Idea is to convert the ASCE network to a logical representation that allows semantic analysis and reasoning about the validity of the case
- Claims are formalized using node properties:
  - ➢ Object O satisfies property P in environment E
- The claim structure is exported as a series of Prolog predicates

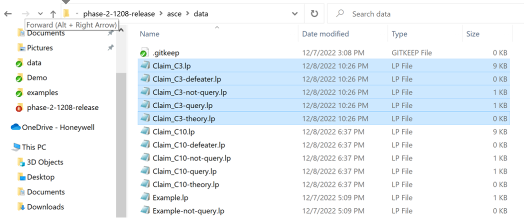
- It is then possible to evaluate the top-level claim as a Prolog *query*. When s(CASP) determines that a claim does not hold, we can determine the reason it does not hold by running the *negated query* and looking at the justification tree to see why the counter-claim holds
- Automated Analysis for:
  - ➢ Semantic-based analysis of CAE
  - ➢ Automatically Identifying Defeaters
  - ➢ Assurance Case Synthesis



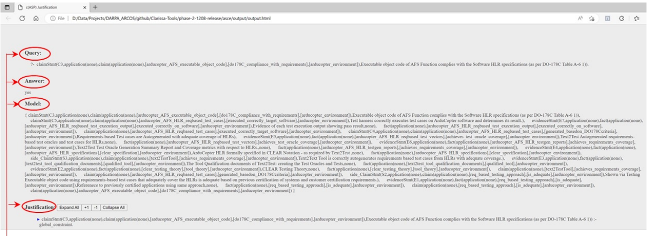


Claim_C3.lp - Notepad

```
claimStmt('C3',application(none),claim(application(none),[arducopter_AFS_executable_object_code],[do178C_com
(as per DO-178C Table A-6 1)'):-
        claimStmt('C5',application(none),claim(application(none),[arducopter_AFS_HLR_reqbased_test_cases],[e
        claimStmt('C4',application(none),claim(application(none),[arducopter_AFS_HLR_reqbased_test_cases],[g
        side_ClaimStmt('S2',application(none),claim(application(none),[req_based_testing_approach],[is_adequ
        claim(application(none),[arducopter_AFS_executable_object_code],[do178C_compliance_with_requirements

claim(application(none),[arducopter_AFS_executable_object_code],[do178C_compliance_with_requirements],[arduc

claimStmt('C5',application(none),claim(application(none),[arducopter_AFS_HLR_reqbased_test_cases],[executed_
        evidenceStmt('E7',application(none),fact(application(none),[arducopter_AFS_HLR_reqbased_test_executi
        claim(application(none),[arducopter_AFS_HLR_reqbased_test_cases],[executed_correctly_target_software

claim(application(none),[arducopter_AFS_HLR_reqbased_test_cases],[executed_correctly_target_software],[arduc

claimStmt('C4',application(none),claim(application(none),[arducopter_AFS_HLR_reqbased_test_cases],[generated
        evidenceStmt('E5',application(none),fact(application(none),[arducopter_AFS_HLR_reqbased_test_vectors
        evidenceStmt('E6',application(none),fact(application(none),[arducopter_AFS_HLR_testgen_reports],[ach
        evidenceStmt('E4',application(none),fact(application(none),[arducopter_AFS_HLR_specifications],[clea
        side_ClaimStmt('S3',application(none),claim(application(none),[text2TestTool],[achieves_requirements
        claim(application(none),[arducopter_AFS_HLR_reqbased_test_cases],[generated_basedon_DO178Ccriteria],

claim(application(none),[arducopter_AFS_HLR_reqbased_test_cases],[generated_basedon_DO178Ccriteria],[arducop

side_ClaimStmt('S2',application(none),claim(application(none),[req_based_testing_approach],[is_adequate],[ar
        evidenceStmt('E1',application(none),fact(application(none),[req_based_testing_approach],[is_adequate
        claim(application(none),[req_based_testing_approach],[is_adequate],[arducopter_environment]).

claim(application(none),[req_based_testing_approach],[is_adequate],[arducopter_environment]).

evidenceStmt('E7',application(none),fact(application(none),[arducopter_AFS_HLR_reqbased_test_execution_outpu
        fact(application(none),[arducopter_AFS_HLR_reqbased_test_execution_output],[executed_correctly_on_so

fact(application(none),[arducopter_AFS_HLR_reqbased_test_execution_output],[executed_correctly_on_software],

evidenceStmt('E5',application(none),fact(application(none),[arducopter_AFS_HLR_reqbased_test_vectors],[achie
        fact(application(none),[arducopter_AFS_HLR_reqbased_test_vectors],[achieves_test_oracle_coverage],[a

fact(application(none),[arducopter_AFS_HLR_reqbased_test_vectors],[achieves_test_oracle_coverage],[arducopte

evidenceStmt('E6',application(none),fact(application(none),[arducopter_AFS_HLR_testgen_reports],[achieves_re
        fact(application(none),[arducopter_AFS_HLR_testgen_reports],[achieves_requirements_coverage],[arduco
```



Name	Date modified	Type	Size
.gitkeep	12/7/2022 3:08 PM	GITKEEP File	0 KB
Claim_C3.lp	12/8/2022 10:26 PM	LP File	9 KB
Claim_C3-defeater.lp	12/8/2022 10:26 PM	LP File	0 KB
Claim_C3-not-query.lp	12/8/2022 10:26 PM	LP File	1 KB
Claim_C3-query.lp	12/8/2022 10:26 PM	LP File	1 KB
Claim_C3-theory.lp	12/8/2022 10:26 PM	LP File	0 KB
Claim_C10.lp	12/8/2022 6:37 PM	LP File	9 KB
Claim_C10-defeater.lp	12/8/2022 6:37 PM	LP File	0 KB
Claim_C10-not-query.lp	12/8/2022 6:37 PM	LP File	1 KB
Claim_C10-query.lp	12/8/2022 6:37 PM	LP File	1 KB
Claim_C10-theory.lp	12/8/2022 6:37 PM	LP File	0 KB
Example.lp	12/7/2022 5:09 PM	LP File	1 KB
Example-not-query.lp	12/7/2022 5:09 PM	LP File	0 KB



s(ASP) Justification
File | D:/Data/Projects/DARPA_ARCOS/github/Clarissa-Tools/phase-2-1208-release/asce/output/output.html

Query:

?- claimStmt(C3,application(none),claim(application(none),[arducopter_AFS_executable_object_code],[do178C_compliance_with_requirements],[arducopter_environment]),Executable object code of AFS Function complies with the Software HLR specifications (as per DO-178C Table A-6 1)).

Answer:

yes

Model:

{ claimStmt(C3,application(none),claim(application(none),[arducopter_AFS_executable_object_code],[do178C_compliance_with_requirements],[arducopter_environment]),Executable object code of AFS Function complies with the Software HLR specifications (as per DO-178C Table A-6 1)), claimStmt(C5,application(none),claim(application(none),[arducopter_AFS_HLR_reqbased_test_cases],[executed_correctly_target_software],[arducopter_environment]),Test harness correctly executes test cases on ArduCopter software and determines its result.), evidenceStmt(E7,application(none),fact(application(none),[arducopter_AFS_HLR_reqbased_test_execution_output],[executed_correctly_on_software],[arducopter_environment]),Evidence of each test execution output showing pass result,none), fact(application(none),[arducopter_AFS_HLR_reqbased_test_execution_output],[executed_correctly_on_software],[arducopter_environment]), claim(application(none),[arducopter_AFS_HLR_reqbased_test_cases],[executed_correctly_target_software],[arducopter_environment]), claimStmt(C4,application(none),claim(application(none),[arducopter_AFS_HLR_reqbased_test_cases],[generated_basedon_DO178Ccriteria],[arducopter_environment]),Requirements-based Test cases are Autogenerated with adequate coverage of HLRs), evidenceStmt(E5,application(none),fact(application(none),[arducopter_AFS_HLR_reqbased_test_vectors],[achieves_test_oracle_coverage],[arducopter_environment]),Text2Test Autogenerated requirements-based test oracles and test cases for HLRs,none), fact(application(none),[arducopter_AFS_HLR_reqbased_test_vectors],[achieves_test_oracle_coverage],[arducopter_environment]), evidenceStmt(E6,application(none),fact(application(none),[arducopter_AFS_HLR_testgen_reports],[achieves_requirements_coverage],[arducopter_environment]),Text2Test Test Oracle Generation Summary Report and Coverage metrics with respect to HLRs.,none), fact(application(none),[arducopter_AFS_HLR_testgen_reports],[achieves_requirements_coverage],[arducopter_environment]), evidenceStmt(E4,application(none),fact(application(none),[arducopter_AFS_HLR_specifications],[clear_specification],[arducopter_environment]),ArduCopter HLR formally specified in CLEAR Notation - as required by Text2Test ,none), fact(application(none),[arducopter_AFS_HLR_specifications],[clear_specification],[arducopter_environment]), side_ClaimStmt(S3,application(none),claim(application(none),[text2TestTool],[achieves_requirements_coverage],[arducopter_environment]),Text2Text Tool is correctly autogenerates requirements based test cases from HLRs with adequate coverage.), evidenceStmt(E3,application(none),fact(application(none),[text2test_tool_qualification_documents],[qualified_tool],[arducopter_environment]),The Tool Qualification documents of Text2Test: creating the Test Oracles and Tests,none), fact(application(none),[text2test_tool_qualification_documents],[qualified_tool],[arducopter_environment]), evidenceStmt(E2,application(none),fact(application(none),[clear_testing_theory],[tool_theory],[arducopter_environment]),CLEAR Testing Theory,none), fact(application(none),[clear_testing_theory],[tool_theory],[arducopter_environment]), claim(application(none),[text2TestTool],[achieves_requirements_coverage],[arducopter_environment]), claim(application(none),[arducopter_AFS_HLR_reqbased_test_cases],[generated_basedon_DO178Ccriteria],[arducopter_environment]), side_ClaimStmt(S2,application(none),claim(application(none),[req_based_testing_approach],[is_adequate],[arducopter_environment]),Shown via Testing Executable object code using requirements-based test cases that adequately cover the HLRs is adequate based on previous certification of systems and customer certification requirements.), evidenceStmt(E1,application(none),fact(application(none),[req_based_testing_approach],[is_adequate],[arducopter_environment]),Refereence to previously certified applications using same approach,none), fact(application(none),[req_based_testing_approach],[is_adequate],[arducopter_environment]), claim(application(none),[req_based_testing_approach],[is_adequate],[arducopter_environment]), claim(application(none),[arducopter_AFS_executable_object_code],[do178C_compliance_with_requirements],[arducopter_environment]) }

Justification:

▸ claimStmt(C3,application(none),claim(application(none),[arducopter_AFS_executable_object_code],[do178C_compliance_with_requirements],[arducopter_environment]),Executable object code of AFS Function complies with the Software HLR specifications (as per DO-178C Table A-6 1)) :- global_constraint.

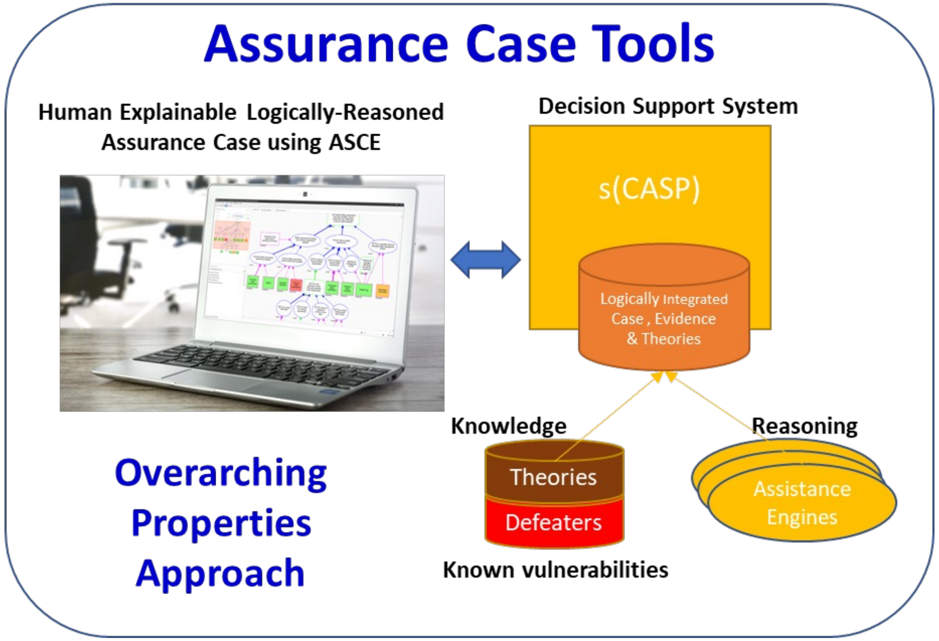
Answer to whether Claim Holds, Resultant Model and Justification

Distribution Statement "A": Approved for Public Release, Distribution Unlimited.
If you have any questions, please contact the DARPA Public Release Center (PRC)

CLARISSA RESEARCH CONTRIBUTIONS

Novelties in Assurance Case Foundations and Realization in Tools

- ❑ Assurance 2.0 methodology ensures focus on positive claims while simultaneously searching for negative defeaters that invalidate the claim
- ❑ Requirements that the completed assurance case should be indefeasible whereby no credible new information would change the judgment i.e., no unresolved doubts
- ❑ Developed "Theory" as reusable assurance case templates with semantics
- ❑ Integrated Assurance for Safety cases and Security cases
- ❑ Automatically translating assurance case as an equivalent logic program that is amenable to common-sense reasoning
- ❑ Automated checks for consistency and completeness
- ❑ Expert user guided synthesis of assurance case with their defeaters


Assurance Case Tools
Human Explainable Logically-Reasoned Assurance Case using ASCE
Decision Support System
s(CASP)
Logically Integrated Case , Evidence & Theories
Knowledge
Reasoning
Theories
Defeaters
Assistance Engines
Known vulnerabilities
Overarching Properties Approach

CLARISSA: Foundations, Tools and Automation Support for Assurance Cases, to appear in DASC 2023 https://2023.dasconline.org/

Distribution Statement "A": Approved for Public Release, Distribution Unlimited.
If you have any questions, please contact the DARPA Public Release Center (PRC)