Two Decades of Assurance Case Tools: A Survey

MIKE MAKSIMOV, NICK FUNG, SAHAR KOKALY, MARSHA CHECHIK

SEPTEMBER 18, ASSURE'18

UNIVERSITY OF TORONTO

McMaster University

Assurance Cases

- Definition – “A reasoned and compelling argument, supported by a body of evidence, that a system, service or organization will operate as intended for a defined application in a defined environment.”

- Often with a particular focus
  - Safety
  - Security
  - Dependability
  - Trust
  - ……

[GSN Standard 2011]

Assurance Cases

We have two types:

- Textual
- Graphical

Within the **context** of the tolerability targets for hazards (from reference Z) and the list of hazards identified from the functional hazard analysis (from reference Y), we follow the **strategy** of arguing over all three of the identified hazards (H1, H2, and H3) to establish sub-claim 1, yielding three additional **claims**: H1 has been eliminated; H2 has been sufficiently mitigated; and H3 has been sufficiently mitigated.

The **evidence** that H1 has been eliminated is formal verification.

The **evidence** that catastrophic hazard H2 has been sufficiently mitigated is a fault tree analysis showing that its probability of occurrence is less than $1x10^{-6}$ per annum. The **justification** for using this evidence is that the acceptable probability in our environment for a catastrophic hazard is $1x10^{-6}$ per annum.

The **evidence** that the major hazard H3 has been sufficiently mitigated is a fault tree analysis showing that its probability of occurrence is less than $1x10^{-3}$ per annum. The **justification** for using this evidence is that the acceptable probability in our environment for a major hazard is $1x10^{-3}$ per annum.

We establish sub-claim (2) within the **context** of the list of hazards identified from the functional hazard analysis in reference Y, and the integrity level (IL) process guidelines defined in reference X. The process **evidence** shows that the primary protection system was developed to the required IL 4. The process **evidence** also shows that the secondary protection system was developed to the required IL 2.

Claim 1: Control system is acceptably safe.
Context 1: Definition of acceptably safe.

Claim 1.1: All identified hazards have been eliminated or sufficiently mitigated.
Context 1.1-a: Tolerability targets for hazards (reference Z).
Context 1.1-b: Hazards identified from functional hazard analysis (reference Y).

Strategy 1.1: Argument over all identified hazards (H1, H2, H3)

Claim 1.1.1: H1 has been eliminated.
Evidence 1.1.1: Formal verification

Claim 1.1.2: Probability of H2 occurring < $1x10^{-6}$ per annum.
Justification 1.1.2: $1x10^{-6}$ per annum limit for catastrophic hazards.
Evidence 1.1.2.: Fault Tree analysis.

Claim 1.1.3: Probability of H3 occurring < $1x10^{-3}$ per annum.
Justification 1.1.3: $1x10^{-3}$ per annum limit for major hazards.
Evidence 1.1.3: Fault tree analysis.

Claim 1.2: The software has been developed to the integrity level appropriate to the hazards involved.
Context 1.2-a: (same as Context 1.1-b)
Context 1.2-b: Integrity level (IL) process guidelines defined by reference X.

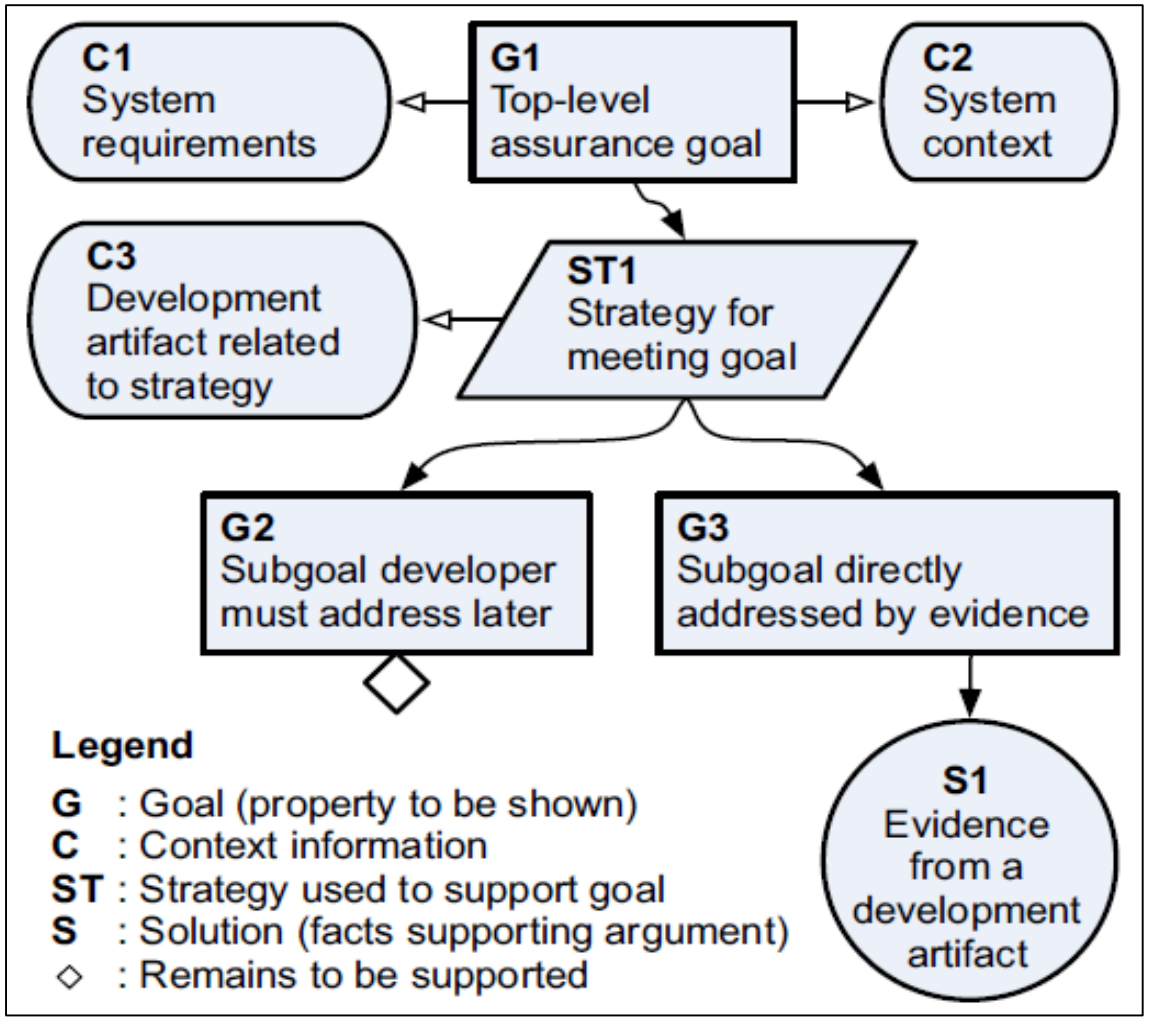
Claim 1.2.1: Primary protection system developed to IL 4.
Evidence 1.2.1: Process evidence of IL 4

Claim 1.2.2: Secondary protection system developed to IL 2.
Evidence 1.2.2: Process evidence of IL 2.

Assurance Cases

We have two types:

- Textual
- Graphical (E.g., Goal Structuring Notation)



Assurance Case Complexity

ACs can grow quite complex in nature

- E.g., an AC for an air traffic control system may comprise over 500 pages and 400 referenced documents

(Lewis, R., Proc. of SSS'09. pp. 183193 (2009))

Assurance Case Tools

Tools aid in creating, maintaining and analyzing ACs

Some notable tools are:

- AdvoCATE (https://ti.arc.nasa.gov/tech/rse/research/advocate/)
- ASCE (https://www.adelard.com/asce/choosing-asce/index/)
- Astah GSN (http://astah.net/editions/gsn)
- CertWare (https://nasa.github.io/CertWare/)
- D-Case Editor (http://www.jst.go.jp/crest/crest-os/osddeos/en/tech.html)
- ISCaDE (http://www.iscade.co.uk/)
- NOR-STA (https://www.argevide.com/home/)
- ……

Motivation

1. Categorize the space of assurance case tools:
   - What AC tools are available and what is state-of-the-art?
   - What are their functionalities and levels of support based on the literature?
   - Are there gaps where further research is necessary?

2. Understand how our tool MMINT-A fits in this space

Presentation Outline

1. Intro and Motivation
2. Search and Evaluation Methodology
3. Results
4. Future Work

Methodology Overview

1. Quasi-Gold Standard (Manual Search)

- Papers that should be in search results.
- Keywords for search

2. Literature Search

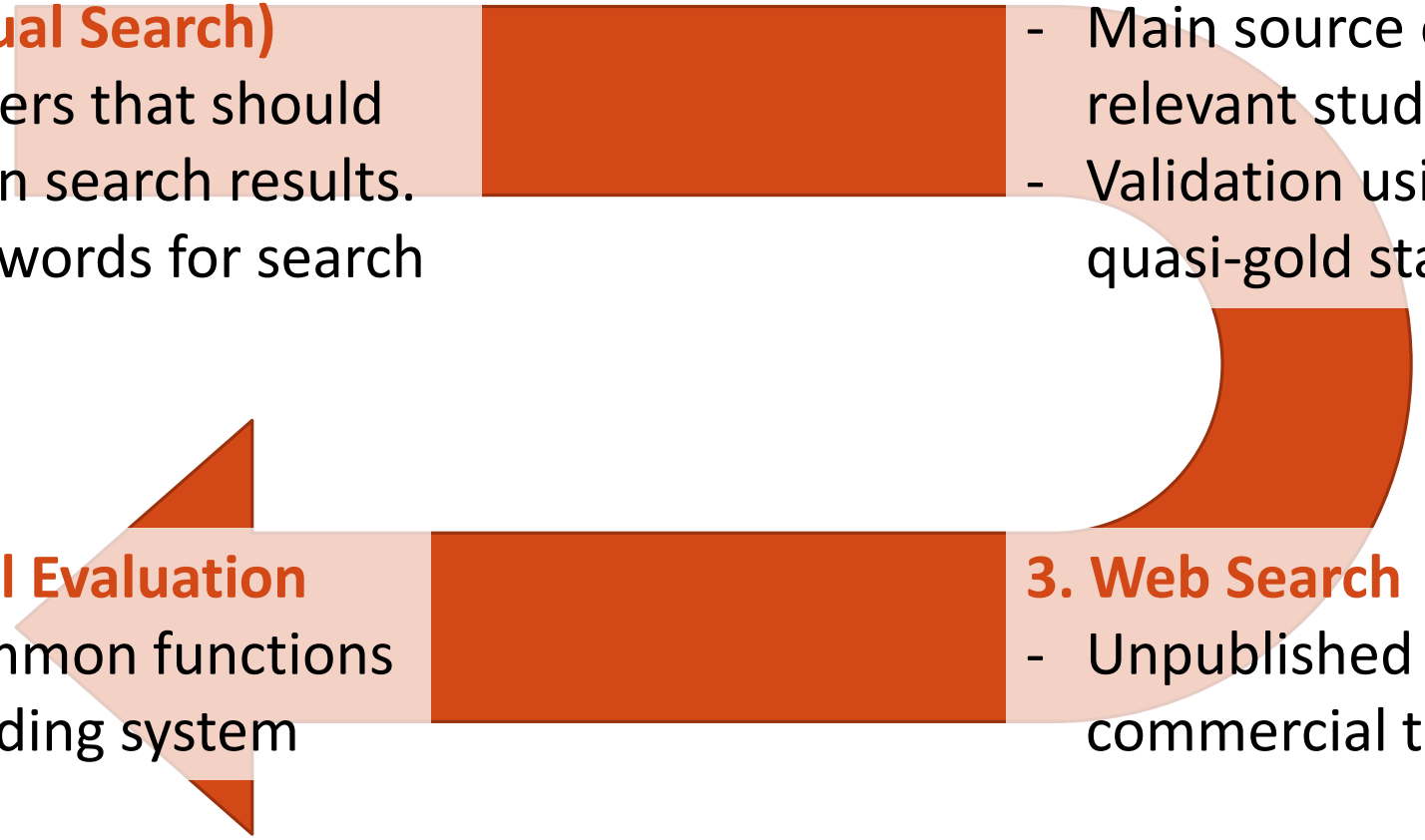
- Main source of relevant studies
- Validation using quasi-gold standard

3. Web Search

- Unpublished and commercial tools

4. Tool Evaluation

- Common functions
- Grading system



Quasi-Gold Standard

Scope:

- 6 relevant conferences and journals (SAFECOMP, HASE, IMBSA, ISSRE, COMPSAC, Reliability Engineering & System Safety)
- 3 years of proceedings (incl. workshops)

Results:

- 10 papers
- 8 keywords related to assurance cases
- 6 keywords related to tools

Search string:

("Assurance Cases" **OR** GSN **OR** SACM **OR** "Safety Case" **OR** "Safety Cases" **OR** "Assurance Case" **OR** "Safety Assurance" **OR** "Safety Compliance") **AND** (Editor **OR** Tool **OR** Editors **OR** Tools **OR** Toolset **OR** Toolsets)

Literature Search

Scope:

- 4 databases (IEEE Xplore, Engineering Village, ACM Digital Library and Springer Link)
- 1998 – 2018

Results:

- 952 papers (80% Quasi-Gold Standard sensitivity)
- 82 relevant and accessible papers
- 38 tools

Web Search

Scope:

- Google
- Top 100 results

Results:

- 8 additional tools (46 in total)

Tool Evaluation

Scope:

- 37 tools with sufficient information
- Evaluation is based on the found literature

Criteria:

- 6 recurring functionalities
  - Creation, Maintenance, Assessment, Collaboration, Reporting, Integration
- 4 grades
  - No Support (D), Minimal Support (C), Moderate Support (B), Strong Support (A)

Tool name	Creation	Maintenance	Assessment	Collaboration	Reporting	Integration
ACBuilder [27]	B	D	D	D	D	D
ACCESS [32]	B	C	C	D	C	D
ACEdit [32]	C	C	B	D	D	D
AdvoCATE [20]	B	B	A	D	A/B	B
AGSN [35]	C	C	B	D	C	D
ASCE [40]	C	B	B	B	A/B	C
Assure-It [45]	C	C/D	D	D	D	D
Astah GSN [32]	B	C	B	D	C	D
Artisan GSN modeler [2]	B	C	B	A	D	D
AutoFOCUS3 [17]	B	B	B	D	D	B
CertWare [13]	B	B	A	C	D	B
D-Case Communicator [38]	C	C	D	C	D	D
D-Case Editor [37]	B	B	B	D	D	B
D-Case Weaver [21]	C	C	C	C	C	B
D-MILS [18]	B	B	B	D	D	B
.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....

Evaluation of capabilities of individual tools

Presentation Outline

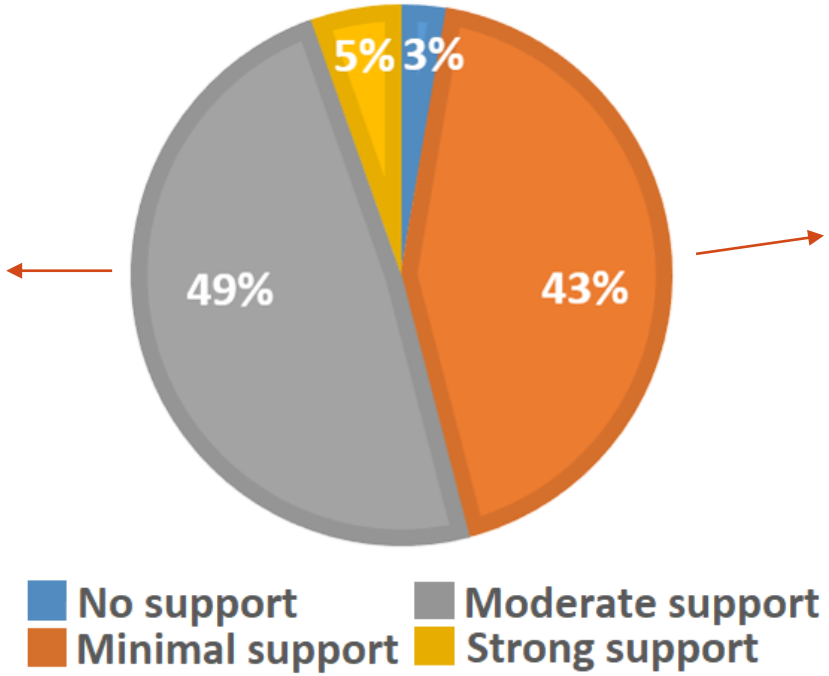
1. Intro and Motivation
2. Search and Evaluation Methodology
3. Results
4. Future Work

Tool Support for Creation

Strong support - Automatic creation of ACs.

No support

Moderate support - Partial automation or reuse in the form of templates/argument patterns.



Minimal support - The user manually creates ACs.

Tool Support for Creation

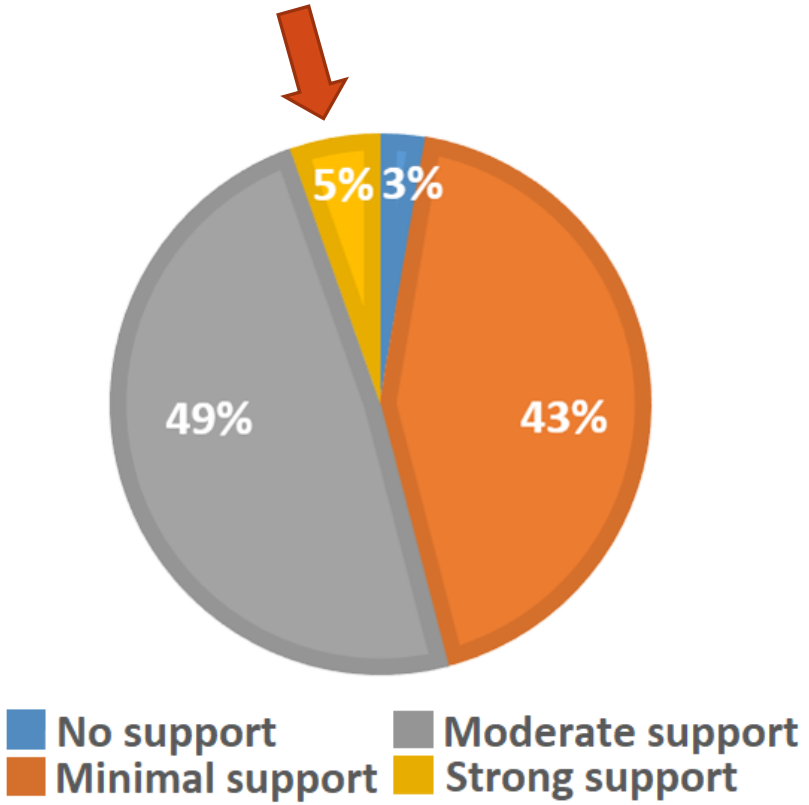
Strong support is offered in the tools:

Resolute (A. Gacek et al.):

- Generates ACs from AADL models
- Limited to distributed embedded systems

ENTRUST (Radu Calinescu et al.):

- Generates ACs from structural and behavioral models
- Limited to self-adaptive software

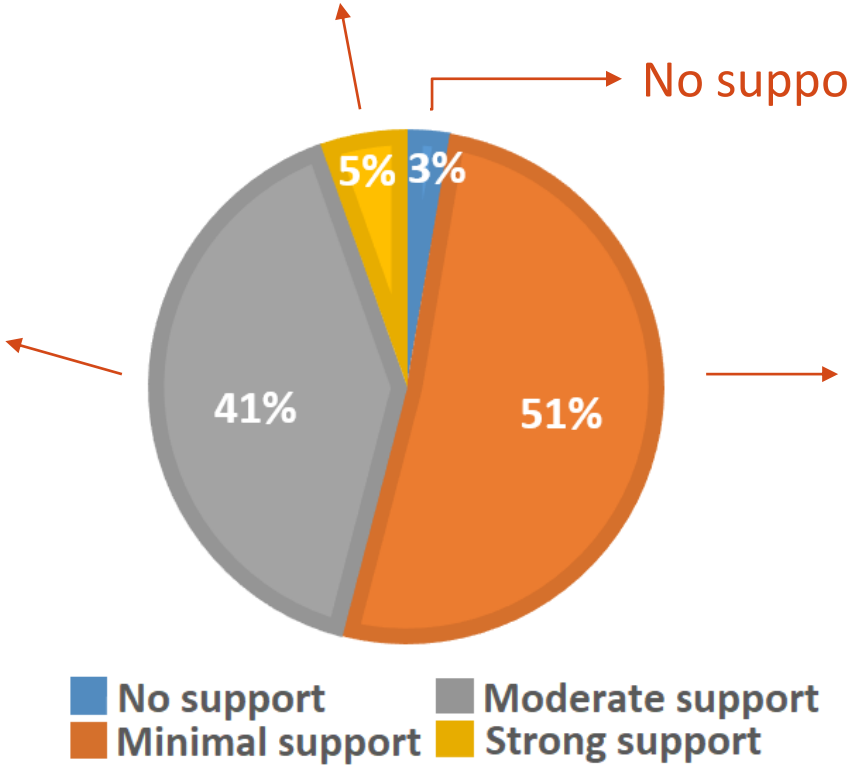


Tool Support for Maintenance

Strong support - Automatic updates of the AC to reflect changes in underlying artefacts (e.g., evidence, system models).

No support

Moderate support - Tracking of relevant artefacts, notifying the user of changes and their impact.



Minimal support – Manual editing with no tool guidance on affected parts.

Tool Support for Maintenance

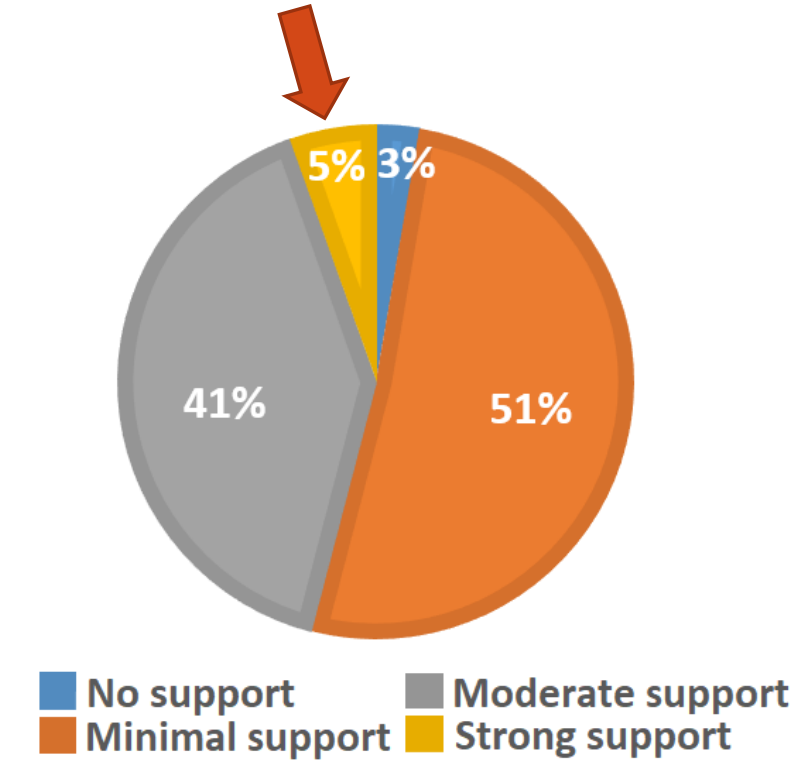
Strong support is offered in the tools:

Evidential Tool Bus (Simon Cruanes et al.):

- Invokes 3rd party tools to generate evidence and re-runs analyses based on outdated evidence

ENTRUST (Radu Calinescu et al.):

- Dynamically verifies self-adaptive systems at runtime and updates the AC accordingly



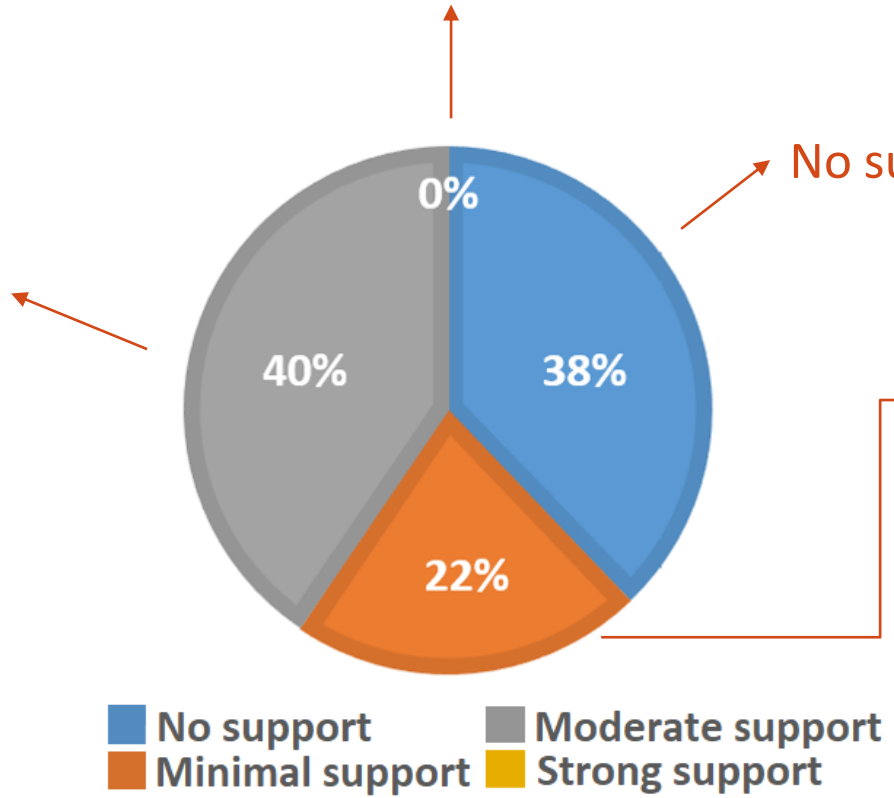
Tool Support for Integration

Strong support – Extensive support for many other design/assurance lifecycle processes.

No support

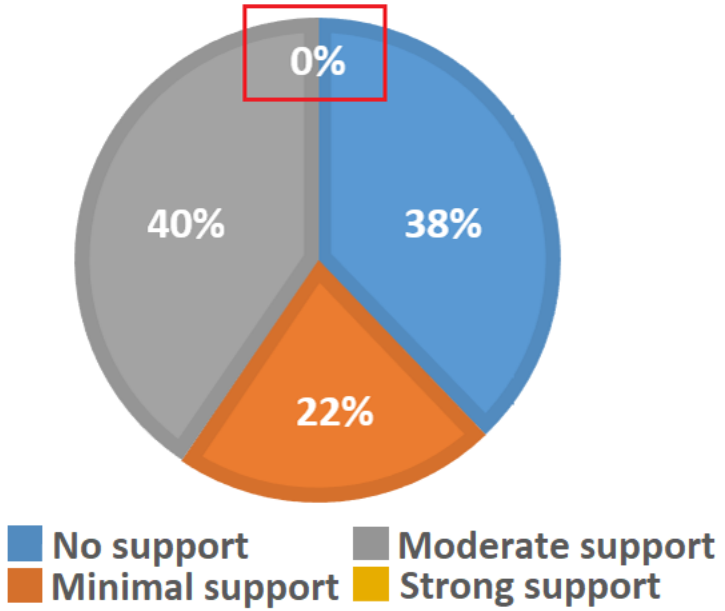
Moderate support – Some support (e.g., bundled with specific 3rd party tool).

Minimal support – Possibility for manual integration with other tools/lifecycle activities.



Tool Support for Integration

- None of the tools offered strong support
- Existing correlation between support for integration, creation and maintenance
- An integrated environment allows coupling between various artefacts, enabling automation through dependencies



Tool Support for Integration

Examples of tools with moderate support:

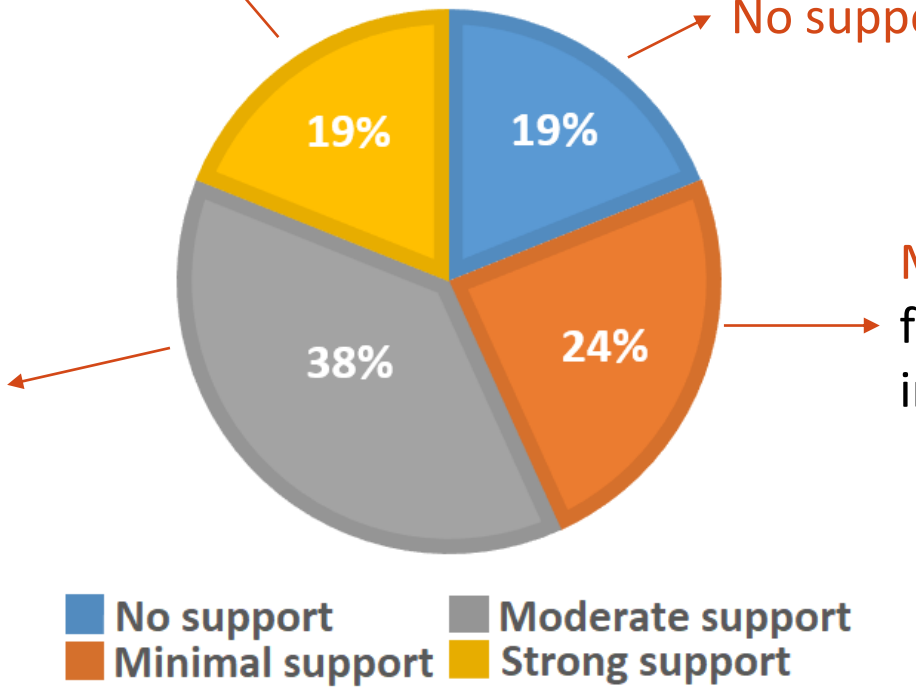
- **AutoFOCUS3 (fortiss):** Requirement models, system models
- **D-Case Editor (Yutaka Matsuno et al.):** Invokes 3rd party verifications tools such as Agda.
- **Resolute (A. Gacek et al.):** AADL system models.
- **ENTRUST (Radu Calinescu et al.):** Structural and behavioral models.
- **AdvoCATE (Ewen Denney et al.):** Invokes 3rd party analysis tools such as AutoCert.
- **TurboAC (GessNet):** Requirement models, test cases, fault tree analysis.

Tool Support for Assessment

Strong support – Syntactic and semantic checks (e.g., validity of the overall argument given its supporting evidence).

No support

Moderate support – Support for syntactic checks (e.g., well-formedness, completeness).



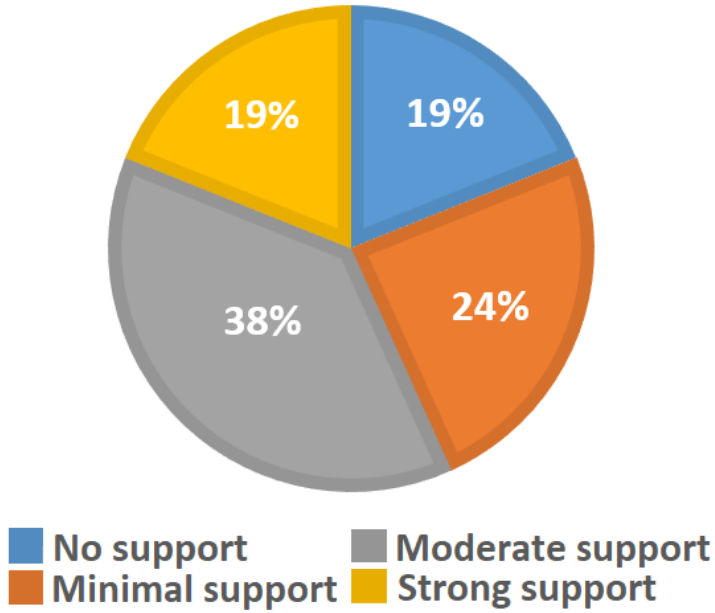
Minimal support – Support for manual annotations to indicate problems.

Tool Support for Assessment

- Highest percentage of strong support

Notable semantic checking approaches:

- Probabilistic reasoning for evidence uncertainty (E.g., ASCE, EviCA, Modus)
- Encoding arguments in a formal checkable language (E.g., D-Case Editor, SafeEd)

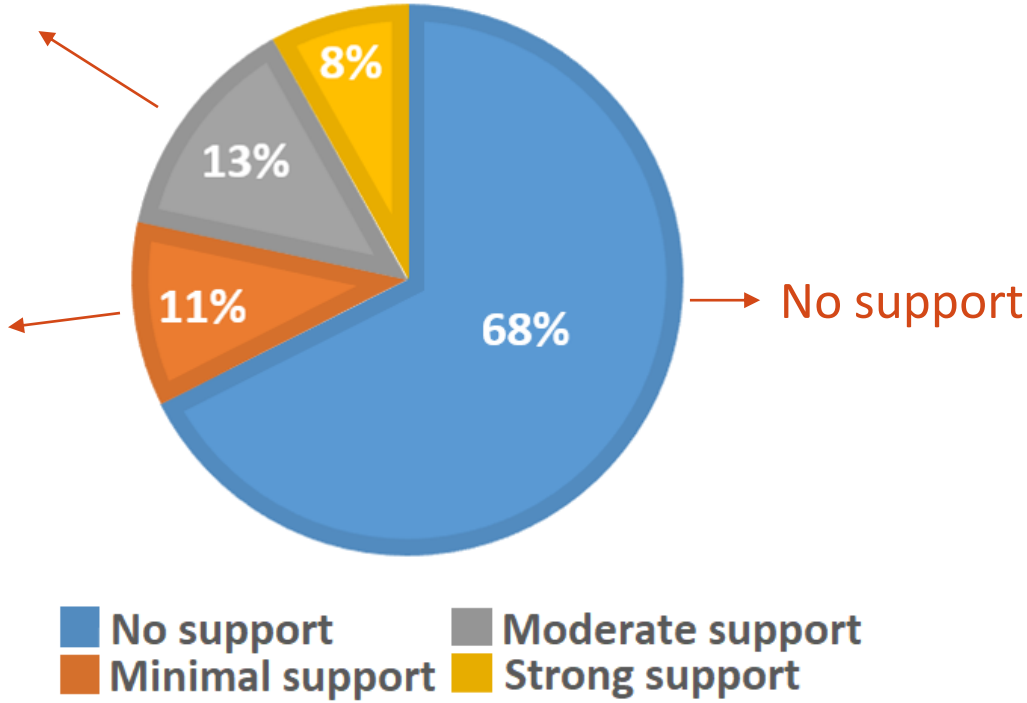


Tool Support for Collaboration

Strong support – Complex multi-user environment (e.g., change requests, reviews and version control).

Moderate support – Additional features such as permission management.

Minimal support – Basic concurrent multi-user environment.



Tool Support for Collaboration

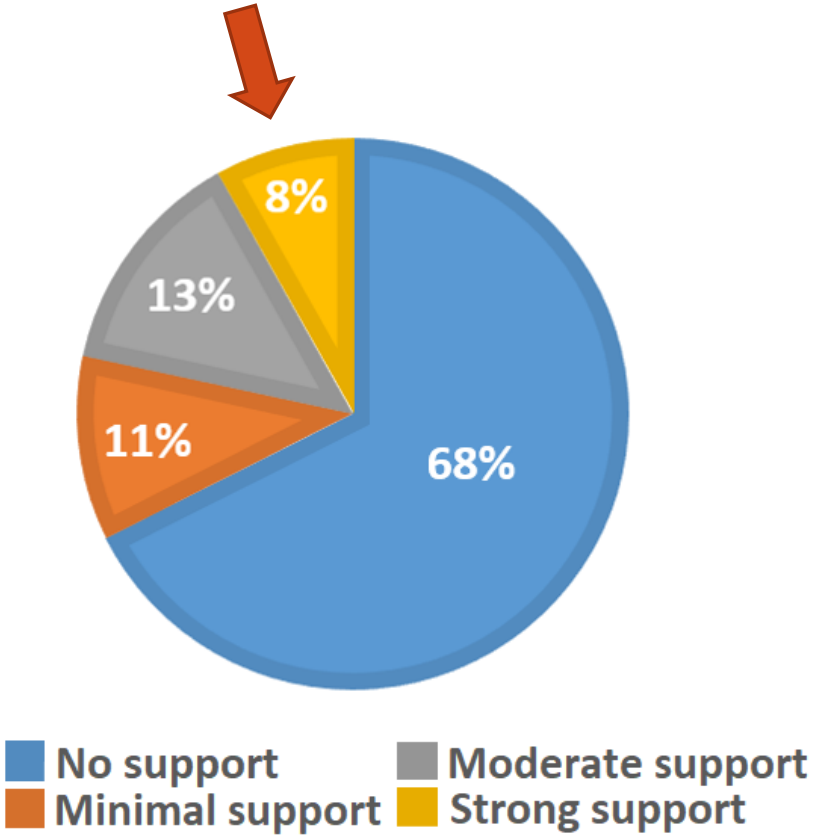
Examples of strong collaboration support:

NOR-STA (Janusz Górski et al.):

- Online multi-user access with permission management
- Integration with internal NOR-STA repositories or user specified ones
- Traceability of all user actions

SCT: Safety Case Toolkit (John Knight et al.):

- Online multi-user access
- Employs Git for version control

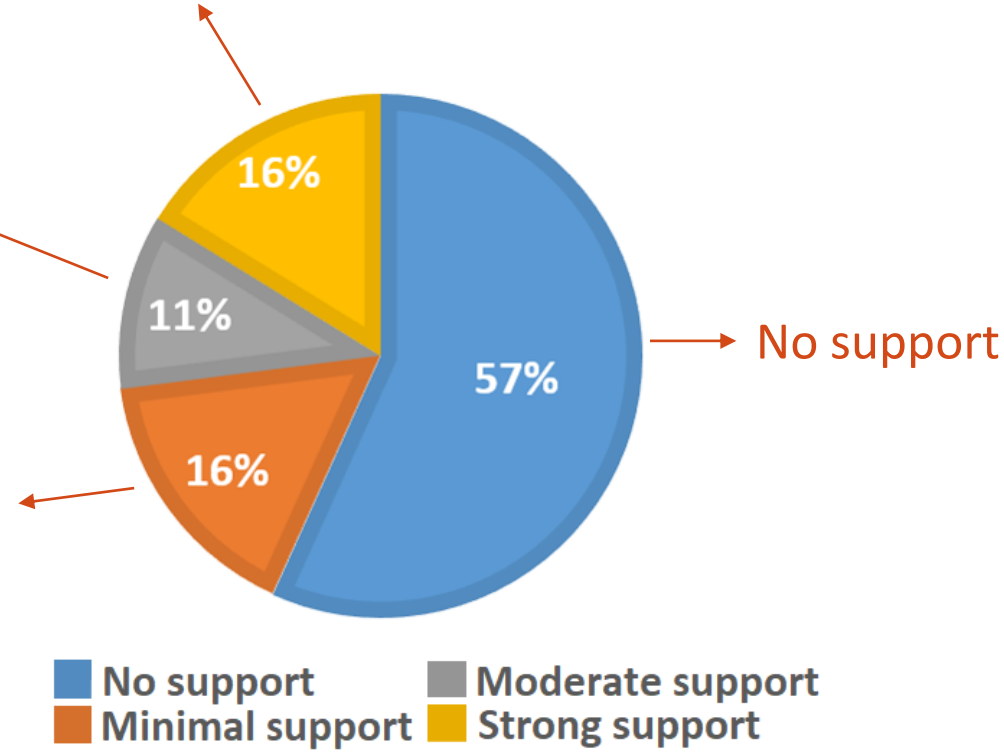


Tool Support for Reporting

Strong support – High user configurability, extensive document formats, and/or detailed/interactive content.

Moderate support – Some user configurability in multiple document formats.

Minimal support – Generic reports, no user configurability, limited document formats.



Tool Support for Reporting

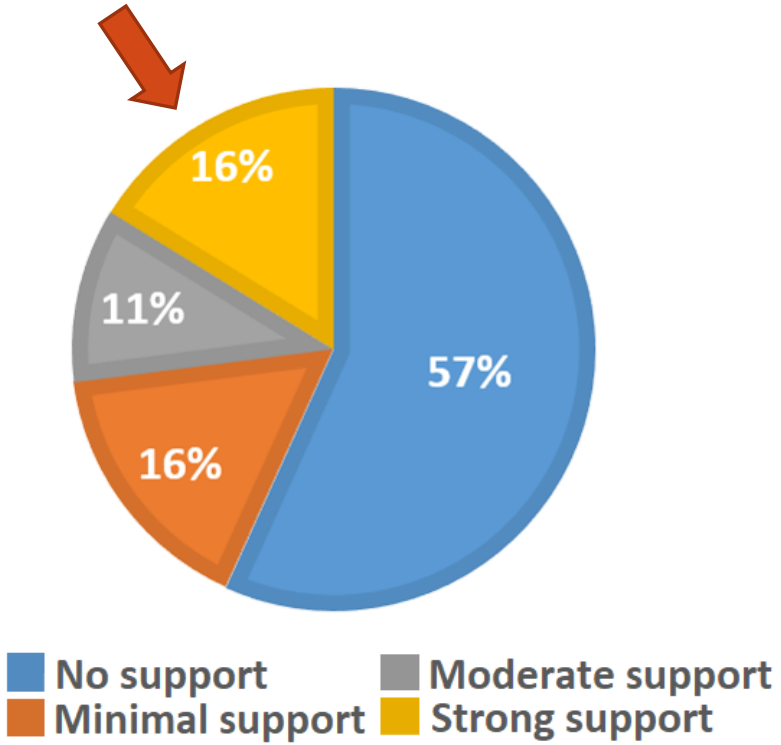
Examples of strong reporting support:

AdvoCATE (Ewen Denney et al.):

- Different reports can be generated from the various views of the tool for different stakeholders
- Offers a number of report templates which are filled based on the user's queries

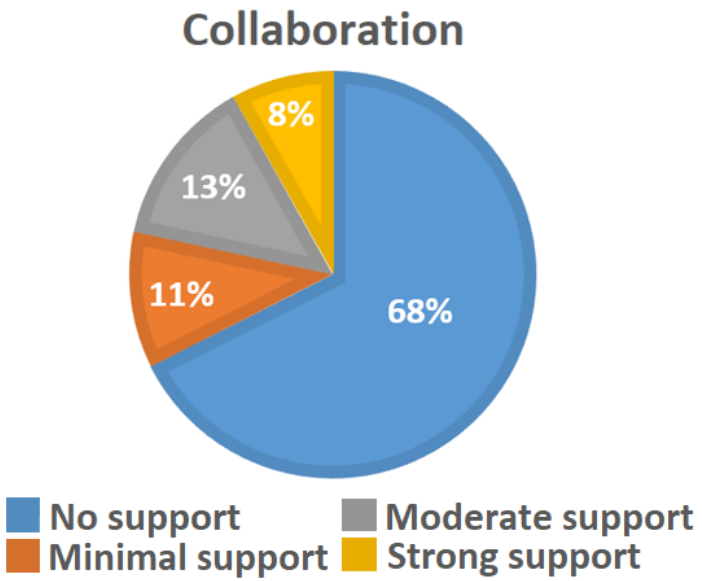
ASCE (Adelard):

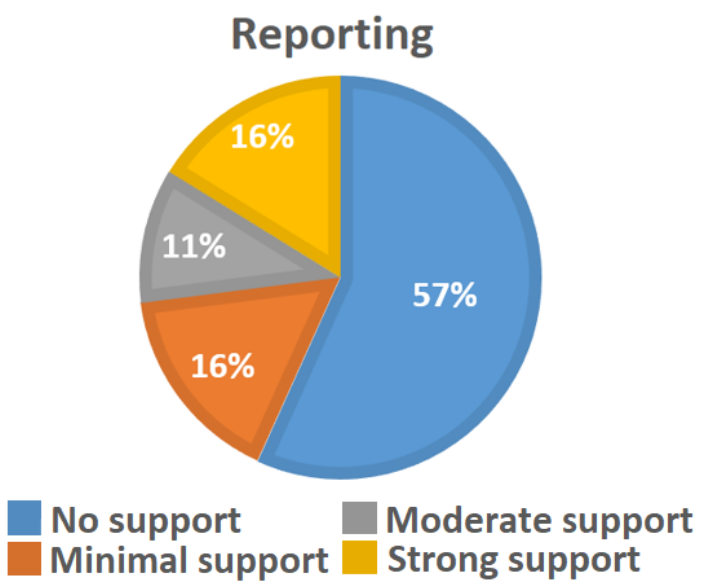
- Standard PDF/Word as well as interactive HTML reports
- Notifies when reports should be updated



Collaboration and Reporting Takeaways

- Large percentage of tools offer no support
- Tools offering these functionalities tend to be industrial/commercial





Other General Findings

- Notations supported:
  - 32 tools (86%) support GSN (Goal Structuring Notation)
  - 3 tools support GSN and SACM (Structured Assurance Case Metamodel)
  - 2 tools support GSN, SACM and CAE (Claims-Arguments-Evidence)
  - 5 tools have their unique notations

Domain-specificity (DS):

- 32 tools are not DS
- 5 tools are DS

(Medical devices, hardware security analysis, reactive embedded software systems, self-adaptive software, real-time embedded systems)

Tool name	Supported notations	Domain
ACBuilder [27]	Textual	Hardware security analysis
ACCESS [32]	GSN	-
ACEdit [32] https://github.com/arapost/acedit	GSN, ARM	-
AdvoCATE [20] https://ti.arc.nasa.gov/tech/rse/research/advocate/	GSN, SACM, Bowtie	-
AGSN [35] https://github.com/AGSNeditor/development	GSN	-
ASCE [40] https://www.adelard.com/asce/choosing-asce/index/	CAE, SACM, GSN, Bowtie	
Assure-It [45]	GSN	-
Astah GSN [32] http://astah.net/download	GSN, ARM, SACM	-
Artisan GSN modeler [2]	GSN	-
AutoFOCUS3 [17] https://af3.fortiss.org/download/	GSN	Distributed, reactive, embedded software systems
CertWare [13] https://nasa.github.io/CertWare/	ARM, CAE, GSN, EUROCONTROL	-
D-Case Communicator [38] https://mlab.ce.cst.nihon-u.ac.jp/dcase/login.html	GSN	-
D-Case Editor [37] http://www.jst.go.jp/crest/crest-os/osddeos/en/tech.html	GSN, SACM	-
D-Case Weaver [21] http://www.jst.go.jp/crest/crest-os/osddeos/en/tech.html	GSN	-
D-MILS [18] https://github.com/phy3rdh/DmilsMBAC	GSN	-
Eclipse & Papyrus extension [26]	GSN	-
eDependabilityCase [33]	GSN	-
ENTRUST [16] https://github.com/gerasimou/ENTRUST	GSN	Self-adaptive software
eSafetyCase Toolkit [41]	GSN	-
.....	.....	.....
.....	.....	.....

General tool information

Summary

- AC creation and maintenance are done completely manually in ~50% of tools
- None of the tools offered strong support for integration
- Integration may allow automation throughout other functional categories
- Over 50% of tools offer no reporting or collaboration capabilities
- GSN is currently the most widespread notation

Presentation Outline

1. Intro and Motivation
2. Search and Evaluation Methodology
3. Results
4. Future Work

Future Work

1. Assurance case tools:
   - Hands-on testing of tools
   - Current area of interest is AC assessment capabilities

     E.g., Syntactic and semantic checks (e.g., validity of the overall argument given its supporting evidence)
   - Compare tools and techniques for AC assessment

2. Compiling a repository of publicly available assurance cases:
   - This repository may serve as a benchmark for testing AC tools.

Questions

Mike Maksimov
maksimov@cs.toronto.edu

UNIVERSITY OF TORONTO