An Artificial Science for System Value Engineering and Assurance

Authors
Department of Computer Science
University of Virginia
Charlottesville, VA 22903
xxxx@virginia.edu

Abstract

abstract goes here

1 Introduction

1.1 Problem

The system shall be adaptable enough to meet new requirements. It shall be resilient enough to continue operating acceptably under unexpected conditions. It shall be easy to use. It shall be scalable.

These and other similarly vague statements about non-functional systems properties (ilities) plague systems engineering today. These statements reflect deeply important concerns, but in ways that are easily misunderstood, not subject to reliable validation, and that cannot support rigorous reasoning, design, and verification. Yet engineering critical non-functional properties is among the most demanding systems engineering challenges.

The underlying problems can be summarized as follows.

1. There are no shared foundational scientific and engineering knowledge, and corresponding precise languages, needed to manage the broad range of non-functional system properties and tradeoffs for complex systems. These properties include changeability, affordability, dependability, usability, resilience, and many more. Gaps in precise, shared understanding of ilities and how they interact in specific environments make it hard to communicate unambiguously about them.
2. There are no validated, precise, reusable framework/template for multi-quality specification. Without such a framework, it would be difficult to reason about the relations and tradeoffs among multiple system qualities when desiging a system.
3. There are no usable science for reasoning about how broad range of strategies impact on and induce tradeoffs among broad range of system qualities. Previous researches focus on discussing the relations (e.g., synergies and conflicts)among the system qualities, but without analyzing the underlying cause of the coresponding relations, and that is the system design strategies. For example, a design strategy, say, hardware redundency could improve one quality (e.g., dependability), but it will also harm another(e.g., cost).
4. There are no unified approach to system specification/prescription, claims/description, evidence, and assurance.

The results of these problems are seen in costly project and operational failures, major down-side surprises late in development, and unacceptable risks, costs, and diffculties in developing and certifying critical systems.

1.2 Purpose of Study

In order to address the above problems, we propose a rigorous, unified, scientific approach to specification, hypothesis, experiment, evidence, and assurance for system qualities and value for cyber-physical-human (CPH) systems. In our approach, we will build a validated, precise, reusable framework for formalizing multi-quality specification. We will promulgate advances of our approach in the form of understandable and usable software tools to enable community evaluation and eventual practical exploitation and evolution of the science in real systems engineering projects. We will also try to clarify nature of system qualities and how they contribute to overall system value.

2 Literature Review

Approaches have been proposed to understanding the ilities. Many of them try to focus on the individual ility, and bring a recogonized understanding of it among the community. For example, Laprie et. al [1] try to summarize the fundamental concepts of dependability, and give definations of it. Knight et. al [2] introduce an operational definition of terms for safety case in the form of a comprehensive certification process for certification based upon the submission of a safety case. Ross et. al [3] have proposed a semantic approach for specifying change-related ility terms, and define a context free language for changeability requirements statements. However, it is far from enough for designing a satisfactory system to just successfully achieve the requirement of one ility but sacrifice the others, because there are normally more than one success-critical stakeholders of each system, and they usually have different ility values, besides, some ileitis conflict with each other, for example, some stakeholders (e.g., end-users) may value the usability most, while some others (e.g., investors) may consider more about the affordability. Therefore, a successful system should achieve a satisfactory balance of ility values for the system's success-critical stakeholders. In other words, when designing a system, we need to consider about all the ilities that the stakeholders value, and reason about tradeoffs between these ilities. Unfortunately, current system acquisition and evolution guidance descriptions have numerous deficiencies and inconsistencies in their coverage of ilities considerations. This situation is becoming more serious as systems and their stakeholders become increasingly complex, dynamic, and diverse [4].

To address the above problem, a few researchers proposed solutions to cover the key ilities of a system at the same time, and reason about the tradeoffs between them. Boehm et al. proposed an initial ontology for reasoning about a system's ilities, in which they provided an elaboration of the needs, a set of initial ilities ontology elements and definitions, examples of their application to some key ilities and their relations [4].

However, in these endeavors the researchers use natural language or quasi-formal definitions, which are often ambiguous and imprecise, to define and explain their theories, which makes their theories difficult to be reliably validated and rigorously reasoned.

Some other researcher are conducting research on assurance case which comprises evidence and argument showing how that evidence supports assurance claims about system ilities (e.g., safety or security). They prefer to use graphical representation and inductive logic to annotate the assurance case of a system, and propose some famous and widely used theory and tools in system design and assurance case analysis, like CAE (Claims-Argumemts-Evidence) [5] and GSN (Goal Structuring Notation) [6]. Graphical representations such as CAE and GSN could help facilitate comprehension and communication of larger arguments they present, but it is doubtful that how well they scale and whether they do help with the evaluation of large assurance cases, because an assuarance case might have large amount of claims and subclaims, along with their arguments, justifications, etc., that make it hard to represent and evaluate the assurance case.

Knight et al. advocate for integrating assurance into the development process by co-developing the software system and its assurance case. Their ABD (Assurance Based Development) approach enables assurance requirements to drive development decisions, and detecting the potential assurance difficulties of a critical system from the earliest stages[7].

Basir et al. propose automtically generate a safety argument by conerting natural deduction style proofs, because they argue that formal proofs generated by automated theorem provers are often complex, machine-oriented, and hard to understand, while "natural dedection style proofs are closer

to human reasoning than resulution proofs". Although they assert that their automatically generated safety argument makes the proofs easier to understand and provide neccessay information to trust the evidence, they show no advantage that their derived argument might have over a hand-generated, informal argument. What's more, they themselves pointed out that the straightforward conversion of ND (Natural Deduction)proofs into safety cases is far from satisfactory as they typically contain too many details and suggest future work on abstraction."[8]

3 Proposed Method

3.1 Technical Approach

We use methods of the sciences of the artificial as an architecture for a unified approach to requirements/specification, system development, claims/hypotheses, experiments and evidence, and assurance, where assurance cases combine mathematically formal system descriptions and theories (including Boehm's taxonomy, for example) with the use of generative skepticism inductive reasoning about evidence.

Coq is expressive enough to embed a vast range of logics and mathematics, including continuous mathematics, making it a candidate framework for specifying, describing, deriving predictions about, and otherwise reasoning about complex systems, including cyber-physical and perhaps cyber-physical-human systems.

3.2 What's new

The novel elements of our approach are as follows.

1. a scientific methods perspective on assurance.

2. an approach unifying scientific theory, skepticism, deductive inference, experiment, evidence, and reasoned judgment about validity.

3. full use of a maximally expressive formal logic notation in assurance cases, consistent with but significantly extending Rushby's use of deductive logic limited to Horn clauses in assurance cases.

4. a formal approach unifying property definition, system prescription (requirements) and system description (claims) following Jackson's ontology, and assurance (arguments).

5. development of a reusable theory of multi-quality system value.

6. instantiation of one lower-level theory for change-related ilities.

7. unification of system qualities under a top-most quality of system value (Boehm work).

8. demonstration of formalization of a reusable template for high-level value specification and assurance cases, leveraging Coq's typeclasses and dependent typing constructs.

9. demonstration of possibility for expanding one of the many qualities in the top level to a next level of granularity, specifically for change-related qualities (CSER paper on our formalization of the Ross model).

3.3 Why it works

Grounding and unifying specification and assurance in the scientific method produces conceptual clarity and justifies the use of deductive logic in this context, strengthening Rushby's separation of logical and epistemic reasoning.

3.4 Technical Hypotheses

4 Technical Evaluation Approach

5 Possible Problems

1. The leap from uncertain evidence to logical axioms involves a kind of round-off error, and these errors will then accumulate up the inference tree, yielding levels of doubt at the root that can be unacceptable even if each individual axiom seems to be strongly, even if not absolutely, justified. For example, 700 claims, each supported by evidence at the level $P = .999$ when conjoined will nevertheless produce $P < 0.5$ at the root of the tree. A question: can one reason effectively about the nature and magnitude of such error? If doubt is quantified probabilistically and inference trees use only ands and ors then yes, methods of fault tree analysis, can be used. But we often lack probabilities for epistemic doubts. We need "pluggable" methods for propagating doubts up trees.
2. Constructive logic and proofs are not understandable and so do not satisfy the human communication requirements for an assurance case

6 Significant of Research

7 Project Plan

8 Conclusion

References

[1] A. Avizienis, J.-C. Laprie and B. Randell: Fundamental Concepts of Dependability. Research Report No 1145, LAAS-CNRS, April 2001.

[2] P. Graydon, J. Knight, and M. Green: Certification and Safety Cases. International System Safety Conference, Minneapolis, MN (September 2010).

[3] Adam M. Ross and D. H. Rhodes: A prescriptive semantic basis for system lifecycle properties, SEAri Working Paper Series, 2012.

[4] B. Boehm, N. Kukreja: An Initial of an Ontology for System ilities. To be presented in 25th Annual INCOSE International Symposium, July 2015.

[5] R. Bloomfield, The Adelard Safety Case Development (ASCAD) Manual. Adelard LLP, London, UK, 1998. Available from: http://www.adelard.com/resources/ascad/

[6] T. Kelly and R. Weaver, "The Goal Structuring Notation: A Safety Argument Notation," Proc. Workshop Assurance Cases: Best Practices, Possible Obstacles, and Future Opportunities, 2004.

[7] Patrick J. Graydon, John C. Knight, and Elisabeth A. Strunk. Assurance-based development of critical systems. In The International Conference on Dependable Systems and Networks, pages 347-357, IEEE Computer Society, Edinburgh, Scotland, June 2007. 55, 61

[8] N. Basir, E. Denney, and B. Fischer: Deriving safety cases from automatically constructed proofs. In: 4th IET International Conference on System Safety, London, UK, The Institutions of Engineering and Technology (2009)

[9] J. Rushby: Formalism in safety cases. In Dale, C., Anderson, T., eds.: Making Systems Safer: Proceedings of the Eighteenth Safety-Critical Systems Symposium, Bristol, UK, Springer (2010) 3-17 1, 3.

[10] J. Rushby, Formalism in safety cases, Making Systems Safer: Proceedings of the Eighteenth Safety-Critical Systems Symposium, Springer, 2010.

[11] J. Rushby, Logic and epistemology in safety cases, In Computer Safety, Reliability, and Security: Proceedings of SafeComp 13, Springer, 2013.

[12] J. Rushby, Mechanized support for assurance case argumentation, in proc. 1st International Workshop on Argument for Agreement and Assurance (AAA 2013), Springer LNCS, 2013.

[13] G. Antoniou, D. Billington, G. Governatori and M. Maher. Representation Results for Defeasible Logic. ACM Transactions on Computational Logic, 2, 255-287, 2001.

[14] M. Maher. Propositional Defeasible Logic has Linear Complexity. Theory and Practice of Logic Programming, 1 (6) 691-711, 2001.

[15] D. Nute. Defeasible Logic. In D. M Gabbay, C. J. Hogger and J. A. Robinson (eds.): Handbook of Logic in Artificial Intelligence and Logic Programming, vol. 3, Oxford University Press, 1994, 353-395.

[16] C.M. Holloway, Safety case notations: alternatives for the non-graphically inclined? In 3rd IET International Conference on System Safety, Birmingham, UK. The Institution of Engineering and Technology, 2008

[17] S. Eagles and F. Wu. Reducing risks and recalls: safety assurance cases for medical devices. Biomedical Instrumentation and Technology 01/2014; 48(1):24-32. DOI: 10.2345/0899-8205-48.1.24