Interlocking Safety Cases for Unmanned Autonomous Systems in Urban Environments

Michael Vierhauser, Sean Bayley, Jane Wyngaard
Department of Computer Science and Engineering
University of Notre Dame
South Bend, IN, USA
{mvierhau,sbayley,jwyngaar}@nd.edu

Jinghui Cheng
Department of Computer Engineering
Polythechnique de Montréal
Montreal, QC, Canada
jinghui.cheng@polymtl.ca

Wandi Xiong, Robyn Lutz
Computer Science Department
Iowa State University
Ames, Iowa, USA
{wdxiong,rlutz}@iastate.edu

Joshua Huseman, Jane Cleland-Huang
Department of Computer Science and Engineering
University of Notre Dame
South Bend, IN, USA
{jhuseman,JaneClelandHuang}@nd.edu

ABSTRACT

The growing adoption of small unmanned aircraft systems (sUAS) for tasks such as eCommerce, aerial surveillance, and environmental monitoring introduces the need for new safety mechanisms in an increasingly cluttered airspace. Safety assurance cases (SAC) provide a state-of-the-art solution for reasoning about system and software safety in numerous safety-critical domains. We propose a novel approach based on the idea of interlocking safety cases. The sUAS infrastructure safety case (iSAC) specifies assumptions and applies constraints upon the behavior of sUAS entering the airspace. Each sUAS then demonstrates compliance to the iSAC by presenting its own (partial) safety case (uSAC) which connects to the iSAC through a set of interlock points. To enforce a "trust but verify" policy, sUAS conformance is monitored at runtime while it is in the airspace and its behavior is described using a reputation model based on the iSAC's expectations of its behavior.

CCS CONCEPTS

• **Software and its engineering** → **Software safety**;

KEYWORDS

sUAS, Unmanned Autonomous Systems, Safety Assurance Cases, Monitoring

ACM Reference Format:
Michael Vierhauser, Sean Bayley, Jane Wyngaard, Jinghui Cheng, Wandi Xiong, Robyn Lutz, Joshua Huseman, and Jane Cleland-Huang. 2018. Interlocking Safety Cases for Unmanned Autonomous Systems in Urban Environments. In *Proceedings of 40th International Conference on Software Engineering Companion, Gothenburg, Sweden, May 27-June 3, 2018 (ICSE '18 Companion),* 2 pages.
https://doi.org/10.1145/3183440.3195035

ICSE '18 Companion, May 27-June 3, 2018, Gothenburg, Sweden

ACM ISBN 978-1-4503-5663-3/18/05.
https://doi.org/10.1145/3183440.3195035

1 INTRODUCTION

The increasing adoption of small unmanned aircraft systems (sUAS) for delivering goods, performing surveillance, conducting search and rescue, and supporting hobbyist activities [1, 5, 7, 10] introduces the need for safety mechanisms to be established at both the infrastructure and the sUAS application level.

The use of sUAS in urban environments meets the definition of a safety-critical system whose "failure could result in loss of life, significant property damage, or damage to the environment" [6]. The problem is a multi-faceted one, in which acceptable levels of safety can only be achieved at the systems level by holistically considering the hardware, software, and operator aspects of the infrastructure and their interactions with potentially untrusted sUAS. We explore safety hazards and mitigations for an urban infrastructure which manages sUAS in the monitored airspace. This includes maintaining awareness of their state, location, and characteristics, and verifying that new sUAS entering the space are capable of meeting minimum safety-related performance requirements.

Our approach uses safety assurance cases (SAC) to describe and reason about system and software safety [9]. SACs provide structured arguments composed of claims, strategies, and evidence that a system is sufficiently safe for use. Evidence is diverse and may include formal models and proofs, simulation results, test cases, and informal artifacts such as compliance with best practices, or training processes. Formal verification has been used successfully in large UAS, for example in the military domain [3, 8]. However it is infeasible to enforce a formal approach upon vendors and operators of diverse sUAS, each of which exhibits different properties.

The initial challenge we seek to address is thus to provide support for software developers as they construct sUAS applications, even though they may lack formal training in safety practices; and furthermore, to assure that their deployed sUAS applications meet satisfactory safety standards. Our approach was evaluated through developing an iSAC and representative uSACs for two emergency response projects, and by using our Dronology system [4] to conduct high-fidelity simulations and outdoor experiments with physical sUAS in order to assess the effectiveness of our monitoring and reputation models.

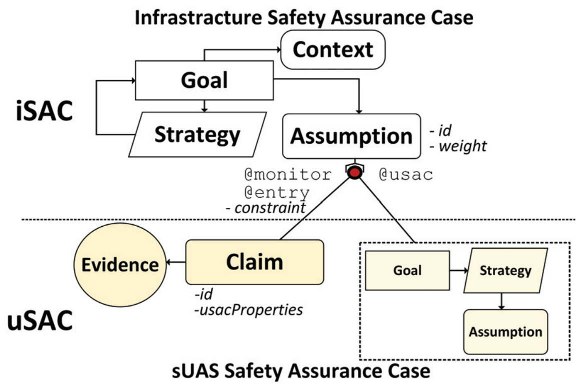

Figure 1: Elements and properties of the iSAC and uSAC.

2 INTERLOCKING ASSURANCE CASES

We propose a novel solution that splits a SAC into two different parts that can be effectively interlocked: an *Infrastructure-level Safety Assurance Case* (iSAC) and *sUAS-level Safety Assurance Cases* (uSAC) for sUAS associated with it. Our approach provides an environment in which the iSAC can be rigorously constructed using proven safety analysis techniques and provides safety guidance to developers constructing sUAS applications. We achieve this by enriching the SAC with annotations defining *interlock points* where potentially untrusted uSACs can interface with trusted and assured iSACs to provide static or runtime checkable constraints. Runtime monitoring is established dynamically as each sUAS approaches the controlled airspace, and is maintained throughout the sUAS' presence within it. If necessary, the infrastructure can respond with runtime, procedural, or even regulatory actions. For example, if the reputation of an sUAS drops below an acceptable level, perhaps because the sUAS fails to consistently report its current position, then the infrastructure could increase the minimum separation distance from other sUAS in the area, and/or could instruct the sUAS to leave the area or even deny it future entry.

The iSAC is created following common SAC construction processes [2, 9]. However, the SAC is extended to facilitate interlocking with the uSAC. To achieve this, *interlock points* are established between sUAS specific *assumptions* in the iSAC and *claims* that the uSAC makes as assertions that it can, and will, meet those assumptions. Our approach includes diverse assumptions which are associated with uSAC claims in three distinct ways: *@entry*, *@monitor*, and *@usac*. The first two represent assumptions associated with monitorable sUAS properties, while the third represents assumptions that are too complex to be fully monitored and therefore require an extension of the SAC. *@entry* represents a property that is monitored one time upon sUAS entry into the space and is transformed into a constraint. Any sufficiently expressive constraint language (e.g., the OCL [11]) can be used to specify and subsequently check conformance to these constraints at runtime. *@monitor* represents a property that is monitored continuously, or on demand, whilst the sUAS is in the controlled area. Finally, *@usac* tags represent assumptions which cannot fully be represented by a simple set of monitorable properties. This could, for example, be due to alternate technologies that might be adopted by different sUAS to achieve the same goal, which could exhibit monitorable properties that are unique to specific design solutions.

Each claim made by the iSAC that is associated with an *@monitor*, *@entry*, or *@usac* assumption must be reflected in the uSAC. Responding to *@monitor* and *@entry* constraints in the uSAC is rather straightforward and typically requires a claim that the sUAS has met the iSAC generated constraint, supported by associated evidence. Responding to *@usac* requests in the uSAC is more challenging. The sUAS developer must provide a full argument demonstrating how the iSAC assumption has been met. Our approach currently does not dynamically analyze pluggable *@usac* arguments, and the uSAC must be provided prior to entry into the controlled airspace. In case of incident (i.e., accident, malfunction, or behavior that deviates from expectations), the full uSAC is available for further evaluation and could be used to hold the sUAS operator accountable.

3 CONCLUSION

We propose a new mechanism for assuring safe use of sUASs in urban environments. The approach uses interlocking safety assurance cases (SACs) to target the intersection of an sUAS infrastructure responsible for controlling an urban airspace and the diverse sUASs that seek to fly in it. By extending safety assurance cases with interlock points, we enable constraints imposed by the iSAC on sUASs entering or operating in its urban airspace to be mapped to evidence of compliance provided by an individual sUAS.

ACKNOWLEDGMENTS

This project has been funded by the Austrian Science Fund (FWF) (J3998-N319) and US National Science Foundation Grants (CCF-1741781, CCF-1649448, and CCF-1513717).

REFERENCES

[1] David J. Anthony, Sebastian G. Elbaum, Aaron Lorenz, and Carrick Detweiler. 2014. On crop height estimation with UAVs. In *Proc. of the 2014 IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*. 4805–4812.
[2] Katrina Attwood, Paul Chinneck, Martyn Clarke, George Cleland, Mark Coates, Trevor Cockram, George Despotou, Luke Emmet, Jane Fenn, Ben Gorry, et al. 2011. GSN Community Standard version 1. *Origin Consulting Limited, UK* (2011).
[3] Julien Brunel and Jacques Cazin. 2012. *Formal Verification of a Safety Argumentation and Application to a Complex UAV System.* Springer, 307–318.
[4] Jane Cleland-Huang, Michael Vierhauser, and Sean Bayley. 2018. Dronology: An Incubator for Cyber-Physical Systems Research. In *Proc. of the 40th Int'l Conf. on Software Engineering: New Ideas and Emerging Results Track (accepted for publication)*.
[5] Milan Erdelj, Enrico Natalizio, Kaushik R. Chowdhury, and Ian F. Akyildiz. 2017. Help from the Sky: Leveraging UAVs for Disaster Management. *IEEE Pervasive Computing* 16, 1 (2017), 24–32.
[6] William S. Greenwell, Elisabeth A. Strunk, and John C. Knight. 2004. Failure Analysis and the Safety-Case Lifecycle. In *Human Error, Safety and Systems Development*. 163–176.
[7] Youngjib Ham, Kevin K Han, Jacob J Lin, and Mani Golparvar-Fard. 2016. Visual monitoring of civil infrastructure systems via camera-equipped Unmanned Aerial Vehicles (UAVs): a review of related works. *Visualization in Eng.* 4, 1 (2016), 1.
[8] Murray L. Ireland, Ruth Hoffmann, Alice Miller, Gethin Norman, and Sandor M. Veres. 2016. A Continuous-Time Model of an Autonomous Aerial Vehicle to Inform and Validate Formal Verification Methods. *CoRR* abs/1609.00177 (2016).
[9] Tim Kelly and Rob Weaver. 2004. The Goal Structuring Notation–a safety argument notation. In *Assurance Cases of Dependable Systems and Networks*.
[10] Jeremy J Kiszka, Johann Mourier, Kirk Gastrich, and Michael R Heithaus. 2016. Using unmanned aerial vehicles (UAVs) to investigate shark and ray densities in a shallow coral lagoon. *Marine Ecology Progress Series* 560 (2016), 237–242.
[11] OMG, Object Management Group. 2014. Object Constraint Language - OCL -Version 2.4. Object Management Group.