Modeling the Safety Architecture of UAS Flight Operations

Ewen Denney, Ganesh Pai, and Iain Whiteside

SGT / NASA Ames Research Center
Moffett Field, CA 94035, USA.
{ewen.denney,ganesh.pai,iain.whiteside}@nasa.gov

Abstract. We develop a notion of safety architecture, based on an extension to bow tie diagrams, to characterize the overall scope of the mitigation measures undertaken to provide safety assurance in the context of unmanned aircraft systems. We use a formal semantics as a basis for implementation in our assurance case tool, AdvoCATE. We also describe the functionality that a safety architecture affords to support both the related safety analysis and subsequent development activities. We motivate the need for a safety architecture through an example based upon a real safety case, whilst also illustrating its application and utility. Additionally, we discuss its role, when combined with structured arguments, in providing a more comprehensive basis for the associated safety case.

Keywords: Argument structures, Bow tie models, Safety architecture, Safety case, Safety system, Unmanned aircraft systems

1 Introduction

Layered or barrier models of safety, as embodied by *bow tie diagrams* (BTDs), have been used in civil aviation for operational safety risk management [1], [2]. The emphasis in this approach is, largely, on maintaining an established safety baseline during operations, and integrating the safety management system (SMS) [3]. BTDs are now also being adopted in the context of regulatory acceptance and operational approval of unmanned aircraft systems (UAS)—our main application domain for this paper—being recommended as the basis for the associated safety case [4], [5].

We have also recently used them in our process for creating real UAS safety cases [6], as part of NASA's UAS traffic management (UTM) effort [7]. Based on that experience, our observation is that the operational focus of BTDs, whilst essential, is insufficient to fully address the different facets of the safety case that must be provided: for example, the assurance concerns pertaining to the changes that may be needed, such as introducing a new technical system that implements a pre-existing safety function, or a new safety function altogether. In this paper, our goal is to provide a more comprehensive basis for a UAS safety case. Our approach is, broadly, to integrate BTDs into our process for safety case/argument development [8].

We have previously explored combining safety assurance arguments with BTDs as a common framework that supports both operational safety assurance [9], as well as pre-operational assurance concerns, such as type design compliance and airworthiness [10].

In the current paper, we build upon our earlier work, although our focus is on extending BTDs in the following ways:

a) identified hazards can be associated with one or more BTDs, each of which can themselves share different admissible BTD elements (described in Section 2). To our knowledge, traditionally there is no representation or means for viewing the full scope of safety concerns. We define a notion of *safety architecture* (SA) to capture this "big picture" of the scope of safety (described in more detail in Section 4).

b) there is a notion of *chaining* BTDs[1] that is loosely related to the above idea of SA. However, so far as we are aware, there is a lack of compatibility rules. As we will see subsequently (Section 3.3), arbitrarily combining certain legitimate BTDs can produce some structures that we may want to reconcile or rule out. Hence, we define structural properties to maintain internal consistency across the whole assemblage of BTDs.

c) we introduce a new notion of bow tie *views* to support specific activities of the safety analysis, assurance, and the subsequent development processes, e.g., risk assessment, specifying barrier functionality, etc. The idea is, in part, to enable BTD use during the pre-operational stages of safety assurance, as well as to facilitate reuse.

d) there is a lack of support for integrating BTDs and assurance arguments within a common safety case.[2] We formalize BTDs and the safety architecture as first-class notions within our toolset, AdvoCATE [12], so as to associate assurance arguments and various elements of the safety architecture.

To the best of our knowledge, none of the commercial BTD tools currently available offer a capability to create SAs, BTD views, or to check their properties.

2 Methodology

Safety risk assessment with the aim of developing a UAS safety case starts with the concept of operations (CONOPS), which describes the intended mission, and the system usage, boundaries, and characteristics. Per NASA program safety requirements [13], we undertake a scenario-based hazard identification to create *hazard risk statements* (HRS). That is, we elaborate the activities, conditions, or entities that pose a potential for harm, specifying the relevant operational context and system state. Then we identify the potential worst-case consequences, after which we undertake a hazard analysis. Traditionally, this is documented in the form of hazard tables, augmented here with BTDs that we incrementally develop, in parallel with the hazard analysis.

Each HRS can be mapped to the *hazard*, and the so-called *top event* and *consequence* elements of a BTD (Fig. 1). From a bow tie perspective, note that hazards capture operational contexts, whereas top events reflect loss of control system states that, if unmitigated, lead to harm. This is compatible with, but subtly different from, hazards as traditionally specified. The events/situations that precede the scenario described by the HRS, traditionally considered as *hazard causes*, correspond to the *threats* leading to a top event in a BTD (Fig. 1).

[1] For example, see http://www.cgerisk.com/knowledge-base/risk-assessment/chaining-bowties/

[2] We are aware of only one other tool that supports both BTDs and argument structures [11].

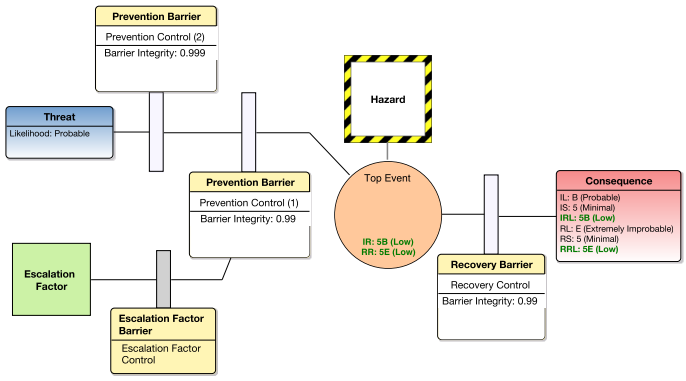

Fig. 1. Simple BTD structure and elements, as implemented in AdvoCATE.

In general, a top event can have a plurality of threats and consequences, although Fig. 1 only illustrates a single threat and consequence for the top event. Intuitively, multiple threats and consequences connected to a single central top event can be seen to resemble a bow tie, giving the structure its name. Also note that a BTD can be viewed as a combination of a fault tree (FT) and an event tree (ET). For example, in one representation, *a*) the left half of a BTD is the FT (rotated right), so that the top event of the BTD is also the top event of the FT, and *b*) the right half of the BTD is the ET so that the top event of the BTD is the *initiating event* of the ET. Other mappings are also feasible [14].

During hazard analysis, we identify pre-existing risk mitigation measures, after which we undertake a risk analysis and assessment towards developing new mitigations. This process iterates until the risk assessment indicates that an acceptable level of safety risk—established on the basis of NASA standards or guidelines, or as per the applicable *Federal Aviation Regulations* (FARs)—will be attained. The collection of mitigation measures, in turn, represent the prevention and/or recovery *barriers* in a BTD (Fig. 1), which will effect risk reduction upon proper implementation. Depending on the level of detail to which we develop the mitigations, we can refine the barriers on a specific path into their constituent *controls*.[3] Further, we can include the *escalation factors* (EFs) which are, effectively, a second level of threats that can compromise barriers. In turn, EFs can be managed by deploying *escalation factor barriers* (EFBs) that are, themselves, identified during a (preliminary or functional) hazard analysis, or through lower-level safety activities.

One of the key outcomes from this process are related BTDs corresponding to the different HRS. Collectively, they give the scope of UAS mission safety, whilst specifying the measures for safety risk mitigation and management. We can see this as a coherent and high-level picture of the overall *safety architecture*, which elaborates how safety is *designed in*, and maintained during operations. As we will describe subse-

[3] The term *barrier* is often used interchangeably with *control* in bow tie literature, although we will distinguish them here.

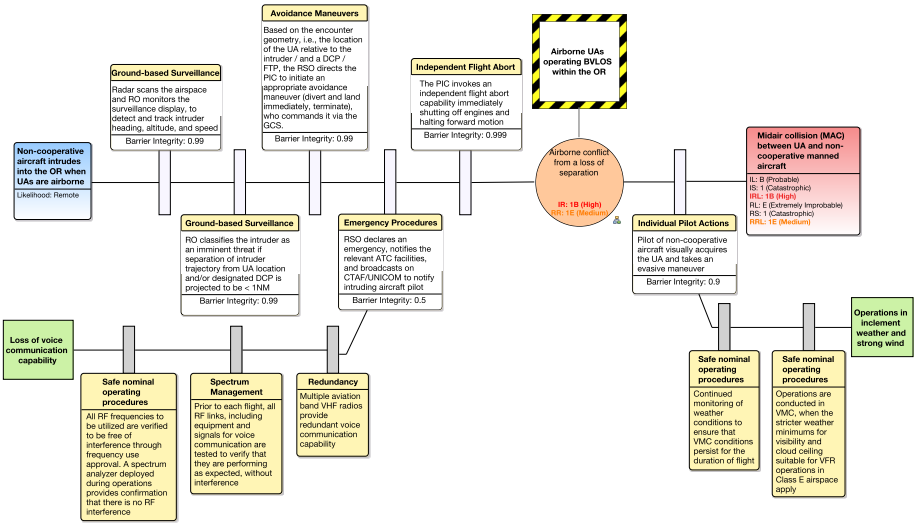

Fig. 2. Fragment of a preliminary BTD for the running example, showing the top event (airborne conflict/near midair collision), an identified threat (intrusion into the OR), and a worst-case consequence (midair collision), together with specific risk mitigation controls, EFs, and EFBs. The specific barriers to which the controls belong have also been given.

quently, this provides a convenient basis for risk assessment, also serving as the interface to address specific assurance concerns. Indeed, the safety architecture together with assurance arguments comprise two key components of a UAS safety case. Although the focus of this paper is on the former, in brief, we apply our methodology for assurance argument development [8] to produce the latter at the level of individual BTDs, as well as for specific bow tie elements, e.g., barriers.

3 Illustrative Example

3.1 Preliminary BTD

In a safety case that we recently developed [6], the CONOPS involves beyond visual line of sight (BVLOS) operations with multiple small UAS, within a defined operating range (OR). The OR is a volume of airspace that includes sparsely populated and minimally built-up areas on the surface, as well as conventionally piloted air traffic (i.e., aircraft with onboard human pilots) within the surrounding airspace. We undertake a hazard analysis on this CONOPS creating BTDs in parallel.

Fig. 2 (made using our tool AdvoCATE) shows a fragment of the preliminary BTD for an identified hazard: *airborne unmanned aircraft (UAs) operating BVLOS within the OR*. An associated top event is *airborne conflict from a loss of separation*. There are other top events (not shown here), such as *deterioration of separation from the terrain*.

A credible worst-case consequence for the identified hazard is a *midair collision (MAC) between a UA and a non-cooperative manned aircraft*. One of the main causes leading to the top event is an airborne intrusion into the OR, which we have shown in Fig. 2 as the threat '*non-cooperative aircraft intrudes into the OR when UAs are airborne*'.

Through the hazard analysis, we identify pre-existing mitigations deployed in the current airspace system that we can use to reduce risk, e.g., pilot actions, such as *see-and-avoid* (as shown in Fig. 2 to the immediate right of the top event). We also develop new mitigation measures, e.g., invoking a flight abort capability that grounds a UA, thereby precluding a near midair collision (NMAC) (shown to the immediate left of the top event in Fig. 2). Fig. 2 also shows some of the identified EFs and corresponding EFBs for the identified barriers. For instance, *loss of voice communication capability* is an EF which, if unchecked, will preclude communication during emergencies, either with air traffic control (ATC), or with other pilots operating in the vicinity. EFBs that contribute to minimizing the associated risk include *redundancy* in the aviation radios used, and *spectrum management* to address potential radio frequency (RF) interference.

In general, the BTDs so created can be interpreted as follows: barriers/controls to the left of the top event represent preventative mitigation measures, while those to the right of the top event are recovery measures to prevent the consequence from occurring.

Also, the visual ordering of barriers/controls corresponds loosely to the temporal order in which they may be invoked, so that they prevent the events preceding them. Thus, events given after a barrier indicate event occurrence after a barrier has been breached. However, the diagrams (intentionally) abstract from the exact ordering and organization of barriers. Indeed, barriers/controls may operate sequentially, in parallel, in a continuous, or a demand mode, etc.

We assume that threats are independently occurring events, i.e., there is a likelihood of simultaneous occurrence and, therefore, they are not disjoint. Consequences can be disjoint events. With this interpretation, threats, top events and consequences can be ascribed an *initial* and a *residual risk level*, computed as a combination of their (initial/residual) *likelihoods* of occurrence and *severity*. Barriers and controls are each ascribed a measure of *integrity*, that corresponds to the likelihood that barriers are breached in a dangerous manner. We use these parameters in safety risk assessment.

3.2 Safety Architecture

We can now expand the scope of the preliminary BTD *a*) to discover opportunities to proactively manage hazards by considering precursors to the identified threats, or *b*) when the controls used may not suffice in terms of their effectiveness.

For example, we can assess the identified threat of Fig. 2 further, by considering it as a top event in a different BTD. Within AdvoCATE, this would create a new, partially developed BTD, where the (erstwhile) threat is now designated as a top event, the top event of Fig. 2 is its consequence, and all the barriers in between the two are retained. Then, we repeat hazard analysis to identify additional threats, mitigating barriers, EFs, and EFBs as appropriate.

Thus, applying our methodology, we incrementally develop BTDs for the different top events for each identified hazard, giving the details of the relevant operational risk

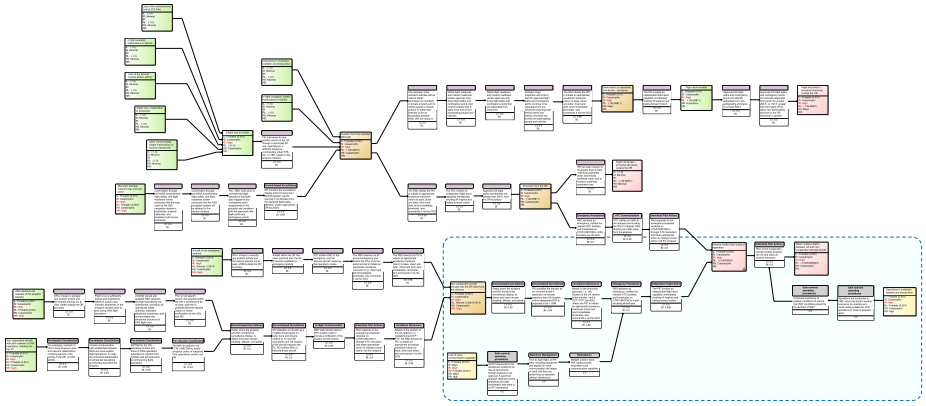

Fig. 3. Fragment of a partially developed safety architecture (SA) (shown zoomed out).

scenarios for the CONOPS, and the applicable safety mitigations. The resulting collection of BTDs comprises the SA for the system, and the overall structure can be seen to characterize the total scope of safety. There is a subtle difference, however, since the SA makes no distinction between top, threat, and consequence events. They are simply events. It is only when we focus on an individual event as the top event of a bow tie that the distinction arises in the resulting BTD.

In our implementation, AdvoCATE automatically assembles the SA in the background as the BTDs are being created. If required, however, the SA can be directly edited and AdvoCATE maintains consistency with the constituent BTDs. Fig. 3 shows a fragment of a (partially developed) SA for our running example. Intuitively, this structure can be considered as a *composition* of related BTDs [9]. As such, we do not distinguish top events from threats or consequences, simply considering event chains and the measures that stop the temporal progression of the associated events. The shaded box to the bottom right shows the part of the SA that corresponds to the BTD fragment shown in Fig. 2; the paths to its left are the events (threats) and controls resulting from further developing the threat identified in Fig. 2, as described earlier in this section.

3.3 Views

When deploying a system that is to be integrated with or without changes to an existing, wider system in which it will be situated, implicit or explicit choices may be required—even during the early stages of development—that affect the SA and, in turn, system safety, e.g., choosing a specific type or number of surveillance sensors, or using specific equipment onboard the airborne system, etc.

The provision of different *views* of the overall SA can well support the associated trade-offs as well as other insights that can be used to refine the SA. Moreover, together with risk assessment, views can aid in providing early assurance that the required safety targets can be met. In turn, that can be used to drive subsequent development stages, e.g., by developing a high-level requirements specification for particular barrier functions. Next, we describe some of the views that we have found useful.

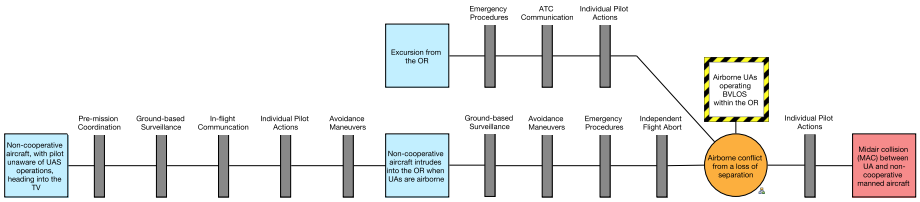

Fig. 4. Barrier-centric view for the running example.

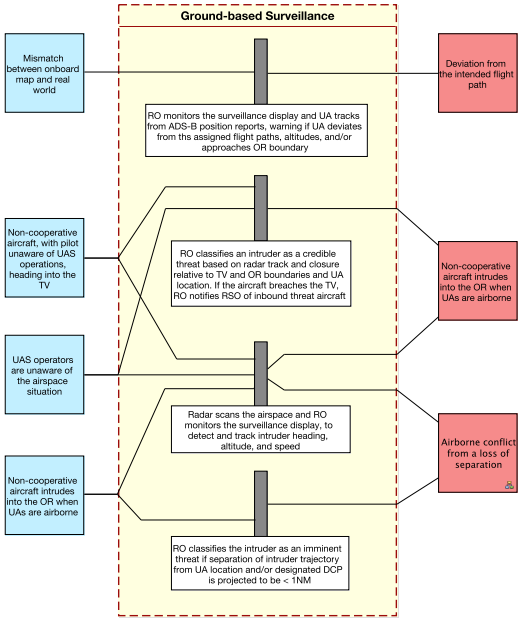

Fig. 5. Slice view of the *ground-based surveillance* barrier.

The *barrier-centric view* of the SA shows a BTD with only the barriers shown, abstracting away the details of the specific constituent controls. Fig. 4 is an example, which includes and expands on the BTD of Fig. 2 (not considering the EFs or EFBs). Specifically, it shows two additional threats, one of which is obtained by further developing the BTD of Fig. 2 as described earlier (Section 3.2).

Although this view abstracts from the details of the controls being used it is useful in a number of ways: *1*) it gives a simple basis for a rapid, albeit qualitative, risk assessment; *2*) it is a higher-level of abstraction at which to apportion risk across the various barriers, given a *safety target*—specified by a regulator, or determined using a guideline such as [5]—and assigned either to the top event or the consequence. That, in turn, provides a reliability (or safety integrity) requirement for barrier design; and *3*) it can be seen as a graphical representation of the traceability from a specific hazard and top event to the barriers used to mitigate their risk.

Another useful collection of views are *slices* relative to the different bow tie elements. For example, a slice focusing on a specific (prevention) barrier gathers all the constituent controls, the threats (or top events) being mitigated and the top events (or consequences) that may result if the barrier is breached.

Fig. 5 shows an example of such a slice across the SA of Fig. 3, focusing on the *ground-based surveillance* barrier. From the perspective of developing and implementing a barrier function, this slice view can be thought of as a system level specification of the required barrier functionality. Moreover, this view presents all the safety concerns being addressed for a specific barrier, and can be useful in communicating to the regulator what a new component of the overall safety system—in this case, ground-based surveillance—is intending to address. Here, note that since the SA of Fig. 3 is partially developed, the barrier slice only shows those controls that have, thus far, been mapped to the barrier, along with the appropriate events being managed. As the SA is developed to completion, AdvoCATE automatically updates the view.

Similarly, a slice focusing on a specific threat can present all the resulting top events and their related barriers. Effectively, that slice is the event chain beginning from a single threat event, across the entire system. Such a view could be useful to focus safety discussions on specific high priority threats presenting all the safety assets available, and how they are organized to manage those threats. A related slice focusing on a consequence shows all the top events leading to a particular consequence/chain of consequences, the associated recovery barriers, and their organization. Thus, we can define other such slices that focus on the different elements (or combination of elements) of the SA, including EFs and EFBs.

3.4 Properties

Whilst creating the SA, it can be useful to highlight potential inconsistencies, and to check that certain[4] well-formedness properties are maintained.

For instance, in incrementally building up the SA, there can be legitimate paths that bypass or *short circuit* controls/barriers [15]. That is, when some controls/barriers on a path are breached, (different) controls (of either the same or different barriers) subsequently used on that path are ineffective. Such structures can result when the composition attempts to reconcile different paths between the same pair of bow tie elements, such that there is at least one path with no other elements between the pair. In the general case, either an intervening independent control/barrier is required on the (short circuiting as well as the short circuited) paths to correct the violation, or there are missing threats in at least one of the BTDs comprising the SA.

Likewise, if there are two paths t—$\mathbf{b}$—c_1 and t—$\mathbf{b}$—c_2, in different BTDs, where t is a threat (or top event), $\mathbf{b}$ represents a collection of barriers (or controls) and c_1, c_2 are different intermediate/top events (or consequences), then there is a potential inconsistency in the overall safety architecture, if there are different outcomes for a common threat t and identical breaches of $\mathbf{b}$. Such a situation may arise if there are missing barriers (or events) on the two paths. In some circumstances, though, we may want to

[4] e.g., Properties whose violations could translate into weaknesses in the risk analysis and, as a consequence, in the implemented safety system.

allow both structures: e.g., when c_1 and c_2 are, in fact, on the same causal chain but the respective BTDs are being applied at different levels of abstraction.

Additional properties that can be checked include structural constraints such as the conditions under which controls can be repeated (e.g., same path, different threats, for a threat and an EF, etc.). In the implementation in our assurance case tool, AdvoCATE, note that we enforce some properties by construction, e.g., loop-freedom and consistency of event ordering, whereas others are permitted with warnings.

4 Formalization

Section 3 gave a worked example illustrating some of the functionality that AdvoCATE supports for BTDs and SAs. We now give the formalization underlying our implementation. First, we note that the structures we want to define are parametrized over underlying sets of events, controls, and barriers.

Definition 1 (Safety Signature). *A* safety signature*, Σ, is a tuple $\langle E, C, B, bar\rangle$, where E, C, and B are disjoint sets of events, controls, and barriers, respectively, and $bar : C \to B$ associates each control with a unique barrier.*

We will henceforth assume the existence of a common safety signature for all definitions. Before defining BTDs formally, we introduce the notion of *controlled event structure* (CES), representing the totality of all events associated with a hazard and their associated barriers and controls. In a sense to be made precise later, a collection of inter-related BTDs specifies this structure. In fact, Fig. 3 shows a CES for one hazard.

Definition 2 (Controlled Event Structure). *A* controlled event structure *(CES) is a tuple $\langle N, \to, l, esc\rangle$ where N is a set of nodes disjoint from the underlying safety signature, $\langle N, \to\rangle$ is a DAG representing temporal ordering of events, l_x, $x \in \{t, d, c\}$ is a family of labeling functions such that $l_d : E \cup B \cup C \to$ string gives descriptions and $l_t : N \to E \cup B$ gives node types. Writing N_e for $\{n \in N \mid l_t(n) \in E\}$ and similarly for N_b, we specify $l_c : N_b \to \mathcal{P}(C)$ such that if $c \in l_c(n_b)$ then $bar(c) = l_t(n_b)$, and $esc : N_b \to \mathcal{P}(N_e)$ such that if $n_e \in esc(n_b)$ then $n_e \to^* n_b$.*

Let $CES(\Sigma)$ denote the collection of CESs over signature Σ. Nodes of a CES represent specific events or barriers (and their associated controls) in the safety system. The same barriers can occur in multiple locations, though each may have different controls (and, as we will see later, can have different integrities). We refer to each such location as a *barrier instance* and likewise for controls.

In contrast to BTDs, which are intended to represent (eventually) well-designed safety systems, a CES models a more general underlying set of events, with a partially developed safety system, possibly without controls yet in place. To facilitate more detailed modeling of a partially developed safety system, we also allow multiple intermediate events between controls, allow events to have multiple successors (i.e., consequences) and predecessors (i.e., causes), so that paths can split and rejoin, and allow empty barriers (for when a particular barrier function is known to be required, before we have chosen or developed its constituent controls).

The interpretation of an escalation branch is that the barriers on the path between the escalation e and barrier b represent escalation factor barriers. Note that the definition allows n-ary escalations, that is, escalations of escalation factor barriers, and so on, though in practice a single level of escalation is typical.

Next, we assign integrity and initial likelihood values to the elements of a CES.

Definition 3 (Initial Risk Assignment). *An* initial risk assignment *for a CES consists of mappings $int_b : N_b \to num$ and $int_c : N_b \times C \to num$, giving integrities of barrier and control instances, respectively, and $lik : N_e \to num$ giving the initial likelihood of event instances.*

We allow separate instances of controls and barriers to have distinct integrities because it is feasible for controls and barriers to have different effectiveness against different threats. Moreover, separate barrier instances can be implemented with different controls. In general, we do not require any consistency between risk assignments for different hazards, since the context is different.

Initial likelihood need only be specified for *global* threats, that is, those that do not have any preceding events. We derive the residual likelihood for all other events (working rightwards). We also assign severity to global consequences (i.e., those that are rightmost), and derive severity on all other events (working leftwards—the simplest approach is to set the severity of an event to be the maximum severity of its immediate successors). The integrity of a barrier is, in principle, derived from its constituent controls. This should be computed via a *risk model*, such as fault trees, possibly implemented via an external tool. Currently, in our tool, we simply provide the integrities of barriers directly and will not consider this further here.

We can now define a safety architecture (SA) as a set of mutually consistent CESs.

Definition 4 (Safety Architecture). *Given safety signature, Σ, a* safety architecture*, $\langle H, l_h, ces \rangle$ consists of a set of hazards, H, hazard descriptions, $l_h : H \to string$, and a mapping $ces : H \to CES(\Sigma)$, which for each hazard, h, returns a controlled event structure such that the CESs are mutually temporally consistent, i.e., for hazards h_1, $h_2 \in H$, if $e \to_1^* e'$ then $e' \not\to_2^* e$.*

Though we require mutual consistency for event ordering (in effect, that the combined relation on events is a DAG), we do not require similar consistency for controls between different CES since they represent different contexts in which controls can be used in different ways. This definition of SA is slightly more permissive than an earlier one given in [9], but is more convenient for implementation and, as we will show below, is essentially equivalent. We now define a BTD as a specific sub-structure of a CES.

Definition 5 (Bow Tie Diagram). *A* bow tie diagram*, B, is a CES such that: 1) there is a designated event, t_p, called the* top event*; 2) there exists at least one* threat*, i.e., an event, t such that $t \to^* t_p$, and one* consequence*, $t_p \to^* c$; and 3) for all events e in B, $e \to^* t_p$ or $t_p \to^* e$, (i.e., all threats lead to a top event, all consequences follow from top events, and all escalations lead to a control that leads to the top event).*

In contrast to the classical notion of BTD, here we permit arbitrary breadth BTDs with intermediate events and arbitrary depth escalations. Moreover, since we allow arbitrary event chains, paths can split and rejoin. A "good" BTD, however, will satisfy

additional properties. For example, it is *free of short circuits* (Section 3.4) if $e \to c \to e' \to c'$ implies $e \not\to c'$. A stronger property that may be enforced is that no barrier can have multiple outputs. We say that a BTD is *maximal* relative to a safety architecture if it includes all events before or after its top event, and *well-controlled* if it has controls between all events. Similar structural conditions that can be indicative of poor design can also be checked.

Next, we show that a CES can be factorized into a set of mutually consistent BTDs which, in combination, give the original CES. To simplify matters, we only consider BTDs relative to a common parent structure, so they can be combined simply by merging. We also assume the existence of initial risk assignments. For shared threats, we must assume that the initial likelihood is the same. Likewise, for severity levels of shared consequences. We need to take care with residual likelihoods since they can differ depending on whether they are computed over paths in a single BTD or the overall CES. However, since those values are derived, we can ignore them when considering the factorization and combination of BTDs.

Theorem 1 (Bow Tie Factorization). *A CES is equivalent to a set of mutually consistent BTDs for distinct top events.*

Proof. Define an ordering on events: $e_1 \leq e_2 \iff \forall e \,.\, e_1 \leftrightarrow^* e \Rightarrow e_2 \leftrightarrow^* e$, where $\leftrightarrow^*$ means comparable (either $\to^*$ or $\leftarrow^*$). Intuitively, e_2 subsumes the set of potential threats and consequences of e_1. It can be seen that $\leq$ is a partial order, so we can talk of maximal elements. Say that maximal events in $\leq$ are *central*. Define a relation, R, on central events, by saying that t_1 and t_2 are *co-central* if they are = relative to $\leq$. Equivalently, $t_1 \; R \; t_2 \iff t_1 \leftrightarrow^* t_2$ and for all events, e, that are not between t_1 and t_2, if $e \to^* t_1$ then $e \to^* t_2$ and if $t_1 \to^* e$ then $t_2 \to^* e$. Since R is an equivalence relation on central events (and a partial equivalence on all events), we can create the partition of central events in the CES by R. Next, choose one member of each partition, and generate the maximal BTD from it. This gives us a set of BTDs. They cover the CES, and can overlap, but are disjoint for central events (and top events, in particular). Since they are sub-dags of the CES they are mutually consistent. □

Thus, a safety architecture can be thought of, equivalently, as a collection of (mutually consistent) BTDs for each hazard.

5 Supporting the Safety Case

Recalling the CONOPS for our running example (Section 2), the safety case is concerned with providing assurance that flight operations can be safely conducted, i.e., that a level of safety can be met that is equivalent to the prevailing safety target with respect to MACs, and that an acceptable level of risk is posed to the population on the ground. Additionally the safety case is required to show that a ground-based detect and avoid capability—comprising radar, transponders, surveillance displays, along with a suite of avoidance maneuvers, and crew functions—can be safely used in lieu of visual observers, the prevailing means of compliance with certain FARs.

Next, we describe the role played by the combination of the SA and argumentation in providing a more comprehensive basis for the safety case. We also discuss the utility of using an SA and its views, and the tool support we have leveraged.

5.1 Assurance of Risk Reduction

We establish that the SA achieves an acceptable residual risk level for the consequence(s), using *1*) barrier integrity, given as nearest order of magnitude estimates based upon data where available and/or conservative assumptions as appropriate; *2*) the (initial) likelihood of occurrence of the threats; *3*) the (initial) severity of the worst-case consequence; and *4*) a *risk model*, based on the barrier-centric view, that combines the above.

Unless using barriers whose specific function is to substantially reduce consequence severity (e.g., frangible airframes), the residual severity for the worst-case consequence is the same as its initial severity. Thus, to assess the magnitude of risk reduction we compute the (residual) likelihood of the consequence(s). We omit the details of the mathematical specification used to implement this capability in AdvoCATE. In brief, however, first we compute top event occurrence likelihood from the initial likelihoods of occurrence of the identified threats, and barrier integrities. Then we compute the likelihood that the consequence in question occurs given the occurrence of the specific top event shown in the view. In either case, the probability of an event (excepting those without any preceding events) is the probability of the disjunction over all paths leading to that event, which we compute using the *inclusion-exclusion principle*. That, in turn, relies on the computation of path probabilities, which is determined for a path as the joint probability that all the events on that path (including barrier breach events) occur. Here, a key assumption is that barrier breaches occur independently.

The safety target against which we compare the residual likelihood is based upon a qualitative risk acceptance matrix [16], i.e., a risk classification. For example, for a midair collision (MAC) consequence (which has a *catastrophic* severity), the safety target is set as at least *extremely improbable*. For other consequences that have different severities, the risk classification, likewise, provides the corresponding safety targets.

5.2 Relationship to Structured Arguments

Whilst safety assurance using risk assessment based on views of the SA can be very useful, it trades off accuracy for simplicity, at the cost of greater uncertainty in the estimate. Although we can usefully improve model accuracy—e.g., considering the specific controls, EFs, EFBs, and by using formalisms such as Bayesian networks, or dynamic event/fault trees [14]—that, itself, can present substantial challenges in quantification and validation [17]. An acceptable compromise is to combine the SA with structured assurance arguments that substantiate specific safety-related assertions, to (qualitatively) offset risk assessment uncertainty. Moreover, argument structures are well suited to supply the rationale why the safety objectives and requirements for changes to a safety baseline—e.g., as derived through a top-down, risk-based approach using BTDs [3]—have been met.

In general, we can associate a number of assurance arguments, each addressing a specific assurance concern, either with a *specific* BTD (or a view), *multiple elements* of

a BTD, or *multiple* BTDs. For instance, we can augment the risk assessment based on the barrier-centric view, as described earlier, with an argument that *1*) not only marshals detailed rationale and evidence substantiating how the barriers (especially those whose integrities are unknown or difficult to quantify) contribute to risk reduction, but also *2*) include a rigorous justification of related assurance concerns such as barrier (failure) independence, sufficiency of the identified threats and event chains, mitigation of common-cause failure modes, etc. Similarly, we can associate arguments with specific barriers [6] (and/or their constituent controls), whose the top-level claims address the appropriate barrier-specific assurance concerns, e.g., provision of the required safety functions, fitness for purpose, achievement of a specific level of safety integrity, etc. At a mission level, assurance arguments straddle the collection of BTDs, justifying why the overall SA enables safety in operation.

5.3 Utility and Tool Support

The SA provides an integrated and consistent view on the full scope of the applicable safety concerns. The barrier-centric view (Section 3.3) supplies the core rationale for how the SA reduces risk: defense-in-depth through independent, loosely-coupled, layers (barriers) of protection. This view also gives a basis for a risk assessment, towards providing assurance of safety during operations. These facets constitute the core value addition provided by the extensions described in this paper, over other approaches.

Additionally, combining assurance arguments and a SA affords a common framework to provide *i*) the pre-operational assurance required for regulatory acceptance of both potential changes to an existing safety system, and the introduction of new safety functions; as well as *ii*) operational safety assurance.

In practice, the tool support that is currently commercially available for creating barrier models[5] largely permits creating only a disconnected collection of BTDs. To the best of our knowledge, they neither support the creation of an SA as we have described it, nor do they provide view-based abstraction. In other words, none of the commercial tools provide the extensions we have developed in this paper. Consequently, there is a need for implementing the associated functionality for it to be useful in practice. The formalization we have described in Section 4 underpins the implementation of BTDs, SAs, views, and the support for risk assessment, in our toolset AdvoCATE. The models for these notions, implemented using the Eclipse Modeling Framework [21], closely follow the formalization we have described.

6 Concluding Remarks

We have described novel extensions to BTDs supporting view-based risk assessment and its integration into an argument-based safety case methodology. We have applied this methodology (including BTD-based risk assessment) in the context of creating, managing, and updating real safety cases required for regulatory approval to conduct BVLOS UAS operations, as part of NASA's UTM effort.

[5] For example, BowTieXP: http://www.cgerisk.com/, BowTie Pro: http://www.bowtiepro.com/, RiskView: http://www.meercat.com.au/, THESIS BowTie: http://www.abs-group.com/, etc.

For instance, in the safety case that provided the running example for this paper, we used views of the underlying SA to communicate to the aviation regulator how safety risk reduction would be achieved during UAS operations. That, in turn, proved to be one of the linchpins of the safety case that contributed to its acceptance and, subsequently, a successful grant of operational flight approval. Our efforts have leveraged our tool, AdvoCATE [12], making full use of its functionality to construct and analyze both BTDs and assurance arguments, including checking properties, creating views, and seamlessly linking and navigating between the two. Our implementation is based on a formal semantics that admits the construction of arbitrary event structures to guide an incremental, interactive development of a well-designed safety architecture.

As mentioned earlier (Section 1), BTDs are used both in civil aviation and for safety assurance of UAS, although existing tools do not implement the functionality we have described here. Moreover, with the exception of one tool that does support argument development [11], so far as we are aware, no other tool provides a common framework to integrate BTDs with assurance arguments for (UAS) safety case development.

Our notion of SA is compatible with classical safety control architectures (i.e., 1oo1, 1oo2, etc.), which represent implementation-level organizations of (largely hardware-based) safety instrumentation and typically exist at a lower-level of abstraction. The work that is, perhaps, most closely related to ours reconciles early architectural knowledge of a system—modeled using AADL—with traditional safety analysis [18]. The focus, however, is on (safety) system design and pre-operational assurance. Our notion of SA is, again, compatible with this work, but conceptually at a higher-level of abstraction. A key point of difference is that our notion retains an operational relevance and, thus, links to the underlying SMS [1].

Our ongoing work is investigating the relationship between argument structures and BTDs both from *a*) the perspective of formal mappings that can be used to generate one from the other, and *b*) how they best complement each other in a safety case. For example, one possibility is to associate argument patterns with generic controls, composing patterns to form an *argument architecture* [9], [19], analogously to how we combine barriers/controls to form the safety architecture. Then, instantiating the argument patterns, based on the context in which the controls are used, will generate the associated assurance argument.

We plan to further develop view generation, leveraging prior work on *queries* [20]. Additionally, we plan to investigate various levels of integration of more detailed quantitative risk analysis models [14], to be able to verify barrier integrity requirements prior to deployment, and to update the risk assessment during operations, based on safety performance monitoring. Eventually, we want to provide capabilities for risk apportionment and deriving the related safety requirements. More broadly, we envision tighter integration into a model-based systems engineering process, with tool support for linking to, and maintaining consistency across, all the safety artifacts relevant for through-life safety assurance.

Acknowledgements. This work was funded by the Safe Autonomous Systems Operations (SASO) project under the Airspace Operations and Safety Program of the NASA Aeronautics Research Mission Directorate.

References

1. FAA Air Traffic Organization: Transforming Risk Management: Understanding the Challenges of Safety Risk Measurement. https://go.usa.gov/xXxea (Dec. 2016)
2. UK Civil Aviation Authority: Bowtie Risk Assessment Models. http://www.caa.co.uk/Safety-Initiatives-and-Resources/Working-with-industry/Bowtie/ (2015)
3. Acfield, A.P., Weaver, R.A.: Integrating Safety Management through the Bowtie Concept: A move away from the Safety Case Focus. In: Australian System Safety Conference. pp. 3–12 (May 2012)
4. Clothier, R.A., Williams, B.P., Fulton, N.L.: Structuring the Safety Case for Unmanned Aircraft System Operations in Non-segregated Airspace. Safety Science, vol. 79, pp. 213–228. (Nov. 2015)
5. Joint Authorities for Rulemaking of Unmanned Systems: JARUS Guidelines on Specific Operations Risk Assessment (SORA) (External Consultation Draft). (Aug. 2016)
6. Denney, E., Pai, G.: Safety Considerations for UAS Ground-based Detect and Avoid. In: 35th IEEE/AIAA Digital Avionics Systems Conference. pp. 1–10 (Sep. 2016)
7. Prevot, T., Rios, J., Kopardekar, P., Robinson III, J., Johnson, M., Jung, J.: UAS Traffic Management (UTM) Concept of Operations to Safely Enable Low Altitude Fight Operations. In: 16th AIAA Aviation Technology, Integration, and Operations Conference. AIAA-2016-3292 (Jun. 2016)
8. Denney, E., Pai, G.: A Methodology for the Development of Assurance Arguments for Unmanned Aircraft Systems. In: 33rd Intl. System Safety Conference. (Aug. 2015)
9. Denney, E., Pai, G.: Architecting a Safety Case for UAS Flight Operations. In: 34th Intl. System Safety Conference (Aug. 2016)
10. Denney, E., Pai, G.: Argument-based Airworthiness Assurance of small UAS. In: 34th IEEE/AIAA Digital Avionics Systems Conference. pp. 5E4-1–5E4-17 (Sep. 2015)
11. Adelard LLP: Assurance and Safety Case Environment. http://www.adelard.com/asce/
12. Denney, E., Pai, G., Pohl, J.: AdvoCATE: An Assurance Case Automation Toolset. In: Computer Safety, Reliability, and Security. SAFECOMP 2012 Workshops. LNCS, vol. 7613, pp. 8–21 (Sep. 2012)
13. NASA Office of Safety and Mission Assurance: NASA General Safety Program Requirements. NPR 8715.3C (Mar. 2008)
14. Dugan, J., Pai, G., Xu, H.: Combining Software Quality Analysis with Dynamic Event/Fault Trees for High Assurance Systems Engineering. In: 10th IEEE High Assurance Systems Engineering Symposium. pp. 245–255 (Nov. 2007)
15. Duijm, N.J.: Safety-barrier Diagrams as a Safety Management Tool. Reliability Engineering and System Safety, vol. 94 (2), pp. 332–341 (Feb. 2009)
16. FAA Air Traffic Organization: Safety Management System Manual version 4.0. (May 2014)
17. Brooker, P.: Air Traffic Management Accident Risk. Part 1: The Limits of Realistic Modelling. Safety Science, vol. 44 (5), pp. 419–450 (Jun. 2006)
18. Feiler, P., Gluch, D., Mcgregor, J.: An Architecture-led Safety Analysis Method. In: 8th European Congress on Embedded Real Time Software and Systems. (Jan. 2016)
19. Denney, E., Pai, G.: Composition of Safety Argument Patterns. In: Computer Safety, Reliability and Security. SAFECOMP 2016. LNCS, vol. 9222, pp. 51–63 (Sep. 2016)
20. Denney, E., Naylor, D., Pai, G.: Querying Safety Cases. In: Computer Safety, Reliability and Security. SAFECOMP 2014. LNCS, vol. 9337, pp. 294–309 (Sep. 2014)
21. D. Steinberg, F. Budinsky, M. Paternostro, and E. Merks, *EMF: Eclipse Modeling Framework 2.0*, 2nd ed. Addison-Wesley Professional, 2009.