TOOLS SUPPORT FOR UAS HAZARD RISK ASSESSMENT

LOUIS DETWEILER, EWEN DENNEY, IAIN WHITESIDE, GANESH PAI
I will cover:

• Definition of UAS
• The history and future of UAS
• Definition and aspects of safety cases
• What AdvoCATE is
• My contributions to AdvoCATE during my internship
WHAT ARE UAS?

- UAS stands for Unmanned Aircraft Systems
- An UAS is composed of several components, including:
  - An UAV (Unmanned Aerial Vehicle)
  - A ground based controller or pilot
  - A communications system between the two
- UAVs are commonly referred to as drones
- UAS are a relatively new, rapidly advancing technology

The Ikhana, a NASA aeronautical research aircraft used to serve the Earth Science community
THE HISTORY OF UAS

- Historically, UAS have mostly been used for military purposes
- The first UAS were developed during and after World War I
- Original uses of these first UAS included using them as targets for training purposes, decoys, and as aerial torpedoes
- Examples of early UAS include:
  - the Ruston Proctor Aerial Target and the Hewitt-Sperry Automatic Airplane (the flying bomb) of 1916
  - The Kettering Bug aerial torpedo of 1918
  - The Larynx early cruise missile of 1927
  - The radio controlled Queen Bee of 1931

Remote piloting of the British “Queen Bee” UAV circa 1935
THE HISTORY OF UAS

- UAS were continually used by militaries around the world after they were first developed.
- During World War II the first UAS operable beyond the line of sight of the ground pilot was developed.
- B-17s were modified into UAS to be used in nuclear tests.
- UAS were also designed to be used as reconnaissance vehicles.
- UAS were first designed to engage in combat in the 1980s.
- UAS play a major role in the wars the US and other countries fight today.

A predator combat drone firing a missile
THE FUTURE OF UAS

With the advancement of UAS technology resulting in decreased cost and required size, their potential for use outside of military applications has grown.

Some theorized applications of UAS include:
- Law enforcement
- Weather research
- Product shipping and delivery
- Aerial photography
- Firefighting
- Agricultural use (crop dusting, etc)
- Passenger transport
- Recreational use (drone racing, etc)

A commercially available recreational/photography UAS
HOW TO GET THERE

• The Federal Aviation Administration (FAA) regulates all aspects of civil aviation

• In order to incorporate the use of UAS into the national airspace, it must be demonstrated that their use can conform to the Federal Aviation Regulations (FARs)

• The main purpose of this requirement is to ensure that UAS operations in the national airspace will be safe

• NASA is currently researching and developing a UAS traffic management (UTM) system that will create airspace integration requirements for enabling safe and efficient UAS operations

Diagram illustrating various civilian applications of UAS in our national airspace
SAFETY CASES

• A safety case is a document that is a structured argument which is supported by evidence that justifies that a given system is sufficiently safe for a specific application in a specific operating environment

• Safety cases identify the hazards and risks associated with the system and describes how the risks are controlled to achieve an acceptable level of safety

• They document the safety management system that ensures the controls are effectively and consistently applied

• Example industries that are regulated using safety cases include the automotive industry, railways, medical devices, aerospace, nuclear power, defense, and aviation

• In particular, safety cases are used to argue that a particular use of UAS will be safe and compliant to FARs
ASPECTS OF SAFETY CASES – GSN ARGUMENT DIAGRAMS

• A key component of safety cases are safety arguments, which are logical arguments that demonstrate how hazards and risks are mitigated in a system.

• These safety arguments can be documented textually, but are often documented graphically in a standardized format called Goal Structuring Notation (GSN).

• GSN uses varying types of nodes to display the argument:
  • Goal nodes which represent things to be proven or shown
  • Strategy nodes which state how goals are to be proven
  • Solution nodes which provide the reasoning that supports the goals
  • Assumption nodes which state any assumptions made in the argument
  • Context nodes which provide contexts for goals or strategies
  • Justification nodes which provide justification for goals or strategies

An example safety argument diagram in GSN form
ASPECTS OF SAFETY CASES – BOW-TIE DIAGRAMS

• Bow-tie diagrams provide a way to document a hazard and their detrimental outcomes, along with safety measures taken to ensure the hazardous event is mitigated

• At the center of the diagram is the hazardous event being considered

• On the left are safety events or threats which could cause the hazardous event to happen

• On the right are the possible detrimental outcomes of the hazardous event

• In between the hazardous event and the threats and outcomes are controls that are put in place to either prevent the threat from causing the hazardous event, or preventing the hazardous event from causing the detrimental outcome

• There are also escalation factors that could contribute to various controls failing
ASPECTS OF SAFETY CASES – HAZARD TABLES

- A hazard table documents all of the potential hazards and risks associated with a given system.
- They generally contain:
  - A list of hazards
  - Hazard causes
  - Hazard consequences and their severities
  - Initial and residual hazard likelihoods and risk levels
  - Requirements for hazard control and mitigation
  - How these mitigations are validated and verified

<table>
<thead>
<tr>
<th>ID</th>
<th>Hazard Risk Statement</th>
<th>Hazard Risk Causes</th>
<th>Hazard Cause Mitigation (Hazard Control Requirement)</th>
<th>Hazard Cause Mitigation Validation</th>
<th>Hazard Cause Mitigation Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Describe the hazard control action plan</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

An example hazard table:

R01: UA exits the operating range and collides with a conventionally piloted (manned) aircraft also operating outside the range, leading to severe injuries and/or fatalities.

- UA exits the operating range and collides with a conventionally piloted (manned) aircraft also operating outside the range, leading to severe injuries and/or fatalities.
- Initial and residual hazard likelihoods and risk levels.
- Requirements for hazard control and mitigation.
- How these mitigations are validated and verified.

---

UTM Shakedown 2 / TCL2 Operational Risks

- Initial and residual hazard likelihoods and risk levels.
- Requirements for hazard control and mitigation.
- How these mitigations are validated and verified.

---

Final Hazard Risk Assessment (FRA): Predicted residual risk level.
ASPECTS OF SAFETY CASES – REQUIREMENT TABLES

A requirement table displays all of the requirements for mitigating hazards in a system and their various attributes.

They generally contain:

- A list of requirements
- The type of each requirement (functional, safety, etc.)
- The sources, or origins, of each requirement
- How the implementation of each requirement is allocated
- The methods by which the requirements are verified
- How the verification methods are allocated
- Indications of the hierarchical structure of the requirements (parent and sub-requirement relationships)
ASPECTS OF SAFETY CASES

- There are more to safety cases than the four aspects previously mentioned, however these four aspects are integral to safety cases and are the ones whose creation tools are currently being developed in...
ADVOCATE – AN ASSURANCE CASE AUTOMATION TOOLSET

- AdvoCATE is a software toolset currently under development at NASA Ames Research Center for the creation and maintenance of safety (assurance) cases
- The goals of AdvoCATE are to reduce the complexity and effort involved in creating safety cases, as well as developing a framework for the automated assembly of safety cases using model based transformations
- In addition to having tools for the manual creation of safety cases, AdvoCATE has a growing number of automated features
- It has capabilities for:
  - including specific metadata
  - translation to and from various formats including other widely used safety case tools
  - safety case composition with automatically generated fragments
  - computation of safety case metrics that allow for a quantitative basis for the assessment of the state of a safety case as it evolves
- During my internship, I contributed to the development of AdvoCATE
ADVOCATE – AN ASSURANCE CASE AUTOMATION TOOLSET

- AdvoCATE is built using the Java programming language, using the Eclipse IDE. In fact, AdvoCATE itself is an Eclipse application.
- AdvoCATE also uses the Sirius Eclipse project which allows for the easy creation of graphical modelling workbenches by leveraging the Eclipse Modeling Framework (EMF) technology.
- Nearly all aspects of AdvoCATE use the Eclipse Modeling Framework as the backing structured data model for the creation of safety case artifacts.
MY CONTRIBUTIONS TO ADVOCATE

- The first few weeks consisted of learning how to use EMF in Eclipse to create and edit domain models, and then using Sirius to present them either graphically or tabularly.

- Next, various validations were added to the argument diagram creation tool of AdvoCATE to:
  - flag errors if there are any cycles in the logic
  - flag warnings if there are any unusual, but sometimes used, connections in the GSN argument diagram.

Examples of my contribution to the argument diagram creator
MY CONTRIBUTIONS TO ADVOCATE

- Some features were also added to the build server that is used to build the AdvoCATE product
- Now every time AdvoCATE is built after changes to the code, the build server runs Checkstyle and FindBugs on the entire code base
- Checkstyle checks the code against predetermined or custom coding style standard, and flags warnings in all the cases where the code differs from that standard
- FindBugs is a static code analyzer that detects possible bugs in code and flags their location with an error marker
MY CONTRIBUTIONS TO ADVOCATE

• Using the Eclipse Nebula NatTable SWT table/grid widget, I created a requirements table editor for AdvoCATE
• The table is kept consistent with a backing EMF data model of the requirements table
• All aspects of the table can be edited, and new entries added, and these changes are propagated to the data model
• There are also sub-tables that show all elements of each type that exist in the table, and these sub-tables are also editable and the changes are propagated to the main table as well as to the data model
**EXAMPLE OF A TABLE BUILT WITH MY EDITOR**

### An example requirements table built in AdvoCATE

<table>
<thead>
<tr>
<th>Description</th>
<th>Source</th>
<th>Implementation Allocation</th>
<th>Allocation</th>
<th>Verification Method</th>
<th>Verification Allocation</th>
<th>Parent</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1 The surveillance system shall detect and track both cooperative and non-cooperative aircraft.</td>
<td>S: Airspace hazard analysis</td>
<td>A: LSTAR Radar based surveillance system</td>
<td>AR1:</td>
<td>VM11: Operational testing - conduct a series of flights within the threat volume and confirm that aircraft equipped with a transponder can be detected and tracked.</td>
<td>VA11: Results of operational testing with transponder equipped aircraft.</td>
<td></td>
</tr>
<tr>
<td>R1.1 The surveillance system shall be capable of receiving transponder signals from transponder equipped aircraft.</td>
<td></td>
<td></td>
<td></td>
<td>VM11: Design verification - confirm that the surveillance system design is equipped with a subsystem for receiving transponder signals.</td>
<td>VA11: Results of preliminary design review (PDR).</td>
<td>R1</td>
</tr>
<tr>
<td>R1.2 The surveillance system shall be equipped with a radar system capable of detecting and tracking aircraft of the size of a single engine Cessna or larger.</td>
<td></td>
<td></td>
<td></td>
<td>VM12: Design verification - confirm that the surveillance system design is equipped with a subsystem for displaying and updating the received transponder signals.</td>
<td>VA12: Surveillance system design artifacts.</td>
<td>R1</td>
</tr>
<tr>
<td>R1.2.1 The surveillance volume shall cover, to the maximum extent possible, the threat volume associated with the operating range.</td>
<td></td>
<td></td>
<td></td>
<td>VM12: Flight testing - conduct a series of flights within the threat volume and confirm that surveillance detects the test aircraft.</td>
<td>VA12:1: Radar acceptance flight test results.</td>
<td>R1.2</td>
</tr>
<tr>
<td>R1.2.2 The surveillance system shall generate a detection for every scan wherein a target has been detected.</td>
<td></td>
<td></td>
<td></td>
<td>VM12: Flight testing - conduct a series of flights within the threat volume and confirm that surveillance detects the test aircraft.</td>
<td>VA12:2: Results of operational testing with transponder equipped aircraft.</td>
<td>R1.2</td>
</tr>
</tbody>
</table>

### An example sub-table of verification methods built in AdvoCATE

<table>
<thead>
<tr>
<th>Description</th>
<th>Verification Method</th>
<th>Verification Allocation</th>
<th>Parent</th>
</tr>
</thead>
<tbody>
<tr>
<td>VM11 Operational testing - conduct a series of flights within the threat volume and confirm that aircraft equipped with a transponder can be detected and tracked.</td>
<td>VA11: Results of operational testing with transponder equipped aircraft.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VM12 Flight testing - conduct a series of flights within the threat volume and confirm that aircraft equipped without a transponder can be detected and tracked.</td>
<td>VA12: Surveillance system design artifacts.</td>
<td></td>
<td>R1</td>
</tr>
<tr>
<td>VM11: Design verification - confirm that the surveillance system design is equipped with a subsystem for receiving transponder signals.</td>
<td>VA11: Results of preliminary design review (PDR).</td>
<td></td>
<td>R1</td>
</tr>
<tr>
<td>VM11: Design verification - confirm that the surveillance system design is equipped with a subsystem for displaying and updating the received transponder signals.</td>
<td>VA12: Surveillance system design artifacts.</td>
<td></td>
<td>R1</td>
</tr>
<tr>
<td>VM12: Design verification - confirm that the surveillance system design is equipped with a subsystem for displaying and updating the received transponder signals.</td>
<td>VA12:2: Results of operational testing with transponder equipped aircraft.</td>
<td></td>
<td>R1.2</td>
</tr>
<tr>
<td>VM12: Flight testing - conduct a series of flights within the threat volume and confirm that surveillance detects the test aircraft.</td>
<td>VA12:1: Radar acceptance flight test results.</td>
<td></td>
<td>R1.2</td>
</tr>
</tbody>
</table>
FUTURE CONTRIBUTIONS

• I will be interning at Ames until May 2018, and will continue working on AdvoCATE

• I will continue adding features to the requirements table editor, and also create a hazard table editor for AdvoCATE
ACKNOWLEDGEMENTS

• I would like to thank my mentors:
  • Ewen Denney
  • Iain Whiteside
  • Ganesh Pai
• NASA Ames Research Center
• NASA Ames Division TI and the Office of Education
• The Oregon Space Grant Consortium