**STEP 5 – Determination of UCAs**

Start at bottom of control structure, this process will do the following, but most will be behind the scenes and the user will be guided without necessarily being aware of the following:

1. Identify hardware and electro-mechanical failures and unsafe interactions
2. Identify UCAs in control structure
3. Identify UCCA for teaming control – macro scale
4. Identify UCA/UCCAs based on role divisions
5. Identify UCA/UCCAs for organizations

**Hardware and Electro-Mechanical Component Analysis**

Analyst should consider the hardware and electro-mechanical failures. This will be further developed as part of causal scenarios but some preliminary evaluation can occur here.

* Probability assessments for hardware and electro-mechanical components (import if known).
* Consider combinations of failures or interactions that can create a hazard:
  + One engine fails catastrophically and that then sends shrapnel that damages other components.
  + A design that results in one component mechanically blocking another component
  + Factors that can affect all components (fuel contamination).

**1. System & Component Overview**  
☐ Review key documentation (schematics, block diagrams, spec sheets)  
☐ List all major components (motors, gearboxes, sensors, actuators, controllers, wiring)

**2. Failure Mode Brainstorm**  
☐ For each component, note common failure types:  
  • Mechanical wear (bearings, belts, gears)  
  • Electrical faults (shorts, opens, insulation breakdown)  
  • Sensor drift or dropout  
  • Control electronics/software glitches

**3. Unsafe Interaction Scan**  
☐ Identify critical interfaces where two or more elements meet (e.g. motor-coupling, sensor-actuator loop)  
☐ Consider how one component’s failure could drive another into a hazardous state (e.g. stalled motor → gearbox overload → housing rupture)

**User should document the findings for each hardware or electro-mechanical component.**

*Control actions from previous step are provided to the user starting at the bottom of the control structure, then branching out as necessary for items on the same level – moving from left to right across that layer, but does* ***NOT*** *consider the ROLES or SHARING of controls. The user does not need to see the control structure here, but the system must ensure that each control previously identified for each controller is presented.*

*User can add additional control actions at this step if they realize they missed one.*

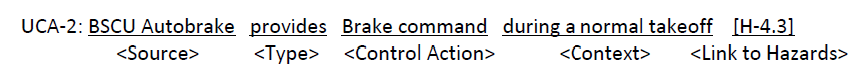
**At each stage the user should be reminded to eliminate items that are outside the scope of the investigation. For example, if applying brakes was listed as a possible control action but the investigation only involves what happens in flight, then brakes should be excluded. The following is incorporated as a reminder each time a UCA is created. The user has the option to exclude it, or add it back later if necessary.**

Tool presents each control action previously identified in step 4 in turn, moving across all of the lowest level of controls.

USER is asked to consider if there is any way that the control action can be unsafe considering each of the following ways that a control action can be unsafe, is there any context where the control action can be unsafe:

* 1. Not providing the control action leads to a hazard.
  2. Providing the control action leads to a hazard (this includes not providing enough or providing too much, for example, a driver might provide too much gas or not enough gas or a pilot might add too much elevator or not enough.
  3. Providing a potentially safe control action but too early, too late, or in the wrong order (think of a driver turning left, right, left instead of right, left, right, as an example of the wrong order).
  4. The control action lasts too long or is stopped too soon (for continuous control

Each UCA consists of five parts (source, type, control action, context and link to hazard – tool to constrain the user such that they complete the UCA correctly. Tagging is automated, UCAs numbered in the order they are analyzed. Following is how a completed UCA should look, without the labels on the second line:



The UCAs are numbered in order, UCA-1, 2, etc.

Expanding on the second line, these are the parts that comprise a UCA

<source controller> <control action> <type, e.g. provides, not provides, too soon/late, too long/short> <context – circumstances that would make this lead to one of the hazards – but not the “why”> <link to at least one of the hazards>.

The parts of the UCA are completed as follows:

UCA is automatically numbered in the order that it was created, UCA-1, 2, 3, etc.

* The system should automatically NUMBER the UCAs as in UCA-1, UCA-2, etc. and automatically connect each one to the Hazard that the user identified. The system does NOT allow the user to complete the UCA unless each part is complete.

***Source*** and ***control action*** were already defined in Step 4 so these are automatically filled in order. So at this point the following is displayed to the user with items to be completed shown in red. (as an example for the pilot-flying (PF) as the controller and “increase thrust” as one of the previously listed controls):

**UCA-1 PF <TYPE> increase thrust <CONTEXT> <LINK TO HAZARDS>**

The system prompts each TYPE of UCA in turn, asking if there is a way that each of these might lead to one of the **previously identified** hazards, asking

**“Would the following result in one of the previously identified hazards?”:**

**NOT PROVIDING** [**fill in control action identified in Step 4]** can be unsafe? YES/NO – if YES, [*under what conditions will that lead to one of the hazards*] *and which hazard will it lead to* (H-1, H-2, etc.)

**PROVIDING** [**fill in control action identified in Step 4**] can be unsafe (consider providing but not enough or too much)? YES/NO - if YES, [*under what conditions will that lead to one of the hazards*] *and which hazard will it lead to* (H-1, H-2, etc.)

**PROVIDING** [**fill in control action identified in Step 4**] **TOO EARLY** can be unsafe? YES/NO if YES, [*under what conditions will that lead to one of the hazards*] *and which hazard will it lead to* (H-1, H-2, etc.)

**PROVIDING** [**fill in control action identified in Step 4**] **TOO LATE** can be unsafe? YES/NO - if YES, [*under what conditions will that lead to one of the hazards*] *and which hazard will it lead to* (H-1, H-2, etc.)

**PROVIDING** [**fill in control action identified in Step 4**] **IN THE WRONG ORDER** can be unsafe? YES/NO - if YES, [*under what conditions will that lead to one of the hazards*] *and which hazard will it lead to* (H-1, H-2, etc.)

**PROVIDING** [**fill in control action identified in Step 4**] for **TOO LONG A PERIOD OF TIME**? YES/NO - if YES, [*under what conditions will that lead to one of the hazards*] *and which hazard will it lead to* (H-1, H-2, etc.)

**PROVIDING** [**fill in control action identified in Step 4**] **TOO SHORT A PERIOD OF TIME** can be unsafe? YES/NO - if YES, [*under what conditions will that lead to one of the hazards*] *and which hazard will it lead to* (H-1, H-2, etc.)

* The tool should provide a PULL DOWN menu of the previously listed hazards so the user can select as many as apply to each hazard. At least ONE must be selected. If NONE are possible, the USER is prompted to consider adding an additional hazard.

In this way each UCA is identified at the basic level. UCAs that have been completed are not displayed, however the user can always pull them up to review them.

**Simultaneous Control Actions – Single Controller**

Consider UCCAs with a single controller. An example is an aircraft pilot with control over pitch, roll, yaw, flaps/slats/spoilers/brakes and thrust. Some of these are more than one control and should be included if applicable. USER should be prompted for:

* Engines [moving thrust separately can be a UCA]
* Brakes [differential wheel braking can be a UCA]
* Other

Independently one of these may be safe, but when conducted in combination with other control actions it can be unsafe. As an example, increased pitch can be unsafe when thrust is not increased.

***Cx is the controller*.** *Control actions are listed here as u1, u2, u3…uN.but are replaced with the actual control actions developed in Step 4 in the UCCA. Tool should INPUT the name of the controller and then list each Control Action identified from Step 4.*

* *Cx* **does not provide {**𝑢1, 𝑢2, 𝑢3…n} **and does not provide** {*u*1, 𝑢2, 𝑢3…n} when… *[H1,2..]*
* *Cx* **does not provide {**𝑢1, 𝑢2, 𝑢3…n} and **provides {**𝑢1, 𝑢2, 𝑢3…n}when… *[H]*
* *Cx* **provides {**𝑢1, 𝑢2, 𝑢3…n} and **provides {**𝑢1, 𝑢2, 𝑢3…n}when… *[H]*
* *Cx* **starts providing {**𝑢1, 𝑢2, 𝑢3…n} before **starting providing {**𝑢1, 𝑢2, 𝑢3…n} when… *[H]*
* *Cx* **starts providing {**𝑢1, 𝑢2, 𝑢3…n} before **ending providing {**𝑢1, 𝑢2, 𝑢3…n} when… *[H]*
* *Cx* **ends providing {**𝑢1, 𝑢2, 𝑢3…n} before **ending providing {**𝑢1, 𝑢2, 𝑢3…n}when… *[H]*

**The next step analyzes the roles which have some similarities to simultaneous control actions.**

Human roles on flight deck with dynamic control sharing

* Considerations given for unsafe interactions.
* Both PF and PM accessing the same controls is considered (such as moving a particular control together, same or opposite direction)
* **Roles addresses when each are using the controls as part of their unique role, but those are working in a way that is unsafe**.
* UCAs established using standard method for each item
* UCCAs established for each control action when combined with the other controller, for example
  + UCCA-1: PM provides flaps too early when PF provides speed reduction too late so flap speed is exceeded [H-3]

*The same level includes the comparison between the same controller but in different roles, i.e., the FO in one role below the Captain, but also the PF in a role below the PM, etc. The roles and control actions are dynamic, and the control actions between them become hazardous only in certain combinations. As there are only two controllers at this level but multiple control actions, the analysis is fairly straightforward.*

* Complete analysis as above for Captain and First Officer and PF and PM. PM is both a control to PF and also takes direction from PF for certain functions. The iteration will result in several additional UCAs, including those of differing roles as well as between the dynamic hierarchies between individuals.
* UCCAs Analyze interactions between PF and PM to identify Unsafe Combinations of Control Actions. Each pilot

*Analyze interactions as separate controllers, each with access to all the controls. The two controllers can be considered the same for this purpose. The context is controller agnostic for this portion of the analysis.*

*Cx is one controller, Cy is the other listed here, one representing PM and the other PF. These are replaced with PM and PF in the written out UCCA that is presented. Control actions are listed here as u1, u2, u3…uN. but are replaced with the actual control actions developed in Step 4 in the UCCA. For the pilots this consists of pitch, roll, yaw, thrust, speed brakes, wheel brakes, flaps, slats, landing gear and flight guidance system (FGS) controls (speed, heading, altitude, engage FMS vertical mode- thrust, pitch, engage FMS lateral mode).*

*Tool automatically swaps whether Cx and Cy represents PF and PM respectively, or vice versa, so all conditions are mapped but conditions that the outcome is agnostic to which is which are only shown once as PF/PM instead of either one. The following are controller agnostic:*

* *Cx* **does not provide {**𝑢1, 𝑢2, 𝑢3…n} and *Cy* **does not provide** {*u*1, 𝑢2, 𝑢3…n} when… *[H]*
* *Cx* **does not provide {**𝑢1, 𝑢2, 𝑢3…n} and *Cy* **provides {**𝑢1, 𝑢2, 𝑢3…n}when… *[H]*
* *Cx* **provides {**𝑢1, 𝑢2, 𝑢3…n} and *Cy* **provides {**𝑢1, 𝑢2, 𝑢3…n}when… *[H]*
* *Cx* **starts providing {**𝑢1, 𝑢2, 𝑢3…n} before *Cy* **starts providing {**𝑢1, 𝑢2, 𝑢3…n} when… *[H]*
* *Cx* **starts providing {**𝑢1, 𝑢2, 𝑢3…n} before *Cy* **ends providing {**𝑢1, 𝑢2, 𝑢3…n} when… *[H]*
* *Cx* **ends providing {**𝑢1, 𝑢2, 𝑢3…n} before *Cy***ends providing {**𝑢1, 𝑢2, 𝑢3…n}when… *[H]*

When there are three or more controllers the analysis is conducted using the following abstraction level.

**In this case controllers are provided in groups for comparison, such as “pilots” or “control room operators”.**

As with the previous UCA’s the tool will automatically number the UCA and will fill in the name of the controller and control action.

* *Any* 𝑐x **does not provide** 𝑢1 and *any other* **does not provide** 𝑐x {𝑢2 or 𝑢3} *when multiple controllers interfere with each other[H]*
* Any 𝑐𝑁 **does not provide** 𝑢1 and *any other* **provides** {𝑢2 or 𝑢3} *when multiple controllers interfere with each other[H]*
* Any 𝑐𝑁 **provides** 𝑢1 and *any other* **does not provide** {𝑢2 or 𝑢3} *when multiple controllers interfere with each other[H]*
* Any 𝑐𝑁 **provides** 𝑢1 and *any other* **provides** {𝑢2 or 𝑢3} *when multiple controllers interfere with each other[H]*
* 𝑐𝑁 **starts providing** 𝑢1 before it **starts providing** {𝑢2 or 𝑢3} when… *[H]*
* 𝑐𝑁 **starts providing** 𝑢1 before it **ends providing** {𝑢2 or 𝑢3} when… *[H]*
* 𝑐𝑁 **ends providing** 𝑢1 before it **starts providing** {𝑢2 or 𝑢3} when… *[H]*
* 𝑐𝑁 **ends providing** 𝑢1 before it **ends providing** {𝑢2 or 𝑢3} when… *[H]*

A screenshot of a diagram

AI-generated content may be incorrect.

**Organizational Analysis – The same as UCCA but organization/department name is substituted at the controller.**

**Last step: User is prompted to filter any remaining UCAs/UCCAs that do not apply**

* CAST analysis: Eliminate UCAs that were not accident factors but not before considering the interaction between apparently unrelated UCAs.
* STPA analysis: Only consider UCAs within the scope of the analysis – but not before considering interaction between apparently unrelated UCAs