DESIGNING SAFE HIGHLY AUTOMATED HUMAN-MACHINE SYSTEMS USING AN STPA-BASED APPROACH: A CASE STUDY

Justin Poh, Nancy G. Leveson
Massachusetts Institute of Technology
Cambridge, MA

Robert R. Copeland
US Army Combat Capabilities Development Command Aviation & Missile Center
Redstone Arsenal, AL

Because modern aircraft are increasingly reliant on automation, it is essential that these aircraft include appropriate human-automation interactions to ensure safe flight. However, it can be challenging to design these interactions using current methods. This paper demonstrates how a design approach based on Systems-Theoretic Process Analysis (STPA) enables more in-depth consideration of human factors engineering and system safety when designing the pilot-automation architecture for an advanced rotorcraft concept. First, STPA is used to derive the control responsibilities needed to safely operate the aircraft. Then, alternative ways to allocate those control responsibilities to the human pilot or automation are compared to identify safety and human factors-related benefits and tradeoffs to help select a preferred pilot-automation architecture. This STPA-based approach thus improves a designer's ability to consider human factors and system safety in an integrated manner and design safety into a system from the beginning.

Although automation can provide useful assistance to a human operator, its use in highly automated systems also changes the roles and responsibilities of the human operator (Leveson, 2011; Wickens, 2004). Instead of having direct control, the operator of a highly automated system might supervise the automation (without being fully aware of how it makes decisions) and may not be able to take over direct control. In addition, they often have limited time to respond to dynamic situations (Flanigen et al., 2022). It can therefore be challenging for them to understand the automation's behavior and respond appropriately if unsafe behavior arises.

For these reasons, the system design must enable safe and effective interactions between human operators and the automation. This requires a design process that integrates human factors alongside other engineering considerations. However, human factors engineering typically occurs separate from the hardware and software logic design (Leveson, 2011). In addition, the roles of the human operator and automation are traditionally defined using levels of automation (Parasuraman et al., 2000; SAE International, 2021). Although these scales provide a general indication of how much automation is used, they do not help to identify the required human-automation interactions or the consequences if unsafe interactions do occur (Copeland, 2019). These categories are also too gross; different types of control might be used in a given situation.

This paper demonstrates a new design process based on Systems-Theoretic Process Analysis (STPA) that makes it easier to integrate human factors and safety considerations into the design process and assists in designing safe and effective human-automation interactions.

An STPA-Based Approach to System Design

STPA is a hazard analysis method that treats safety as a control problem. It emphasizes the need for adequate control over the behavior of the system to ensure properties such as safety are achieved (Leveson & Thomas, 2018). STPA interprets this concept of control broadly. The controls could be technical, physical, social, or organizational. Thus, by analyzing a system using STPA early in the design process, the results can inform downstream design decisions while accounting for human factors and safety considerations in an integrated manner.

STPA models the controls in a system using a hierarchical control structure comprised of a controlled process and controllers (Leveson & Thomas, 2018). A controller (which may be automated or human) enforces the safety constraints by applying appropriate control actions and receiving feedback about the effect of those controls. See Figure 1. Each controller also includes a process model (mental model for humans) that enables appropriate control action selection.

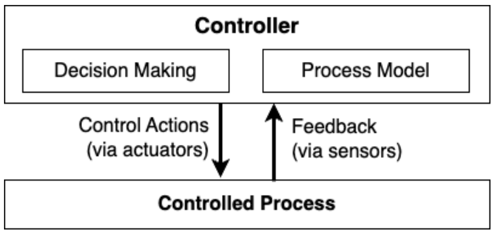

Figure 1. A Simple Control Loop (Leveson & Thomas, 2018)

In (Poh, 2022), a structured STPA-based process was developed for creating a system architecture. See Figure 2. The process begins with an STPA analysis of the system to determine how unsafe behavior might occur. System requirements are then derived to prevent those unsafe behaviors from occurring. Then, the system-level control behavior needed to meet those system requirements is defined by identifying four types of control elements: (1) Responsibilities (a function to be performed), (2) Process model parts, (3) Feedback and (4) Control actions. Timing requirements are also defined to specify the speed and frequency with which the responsibility must be carried out. Finally, architecture options are created by assigning the responsibilities and their associated control actions and feedback to the human or engineered components in a system. Each architecture option can then be further analyzed using STPA and the results compared to determine the safety-related benefits and tradeoffs.

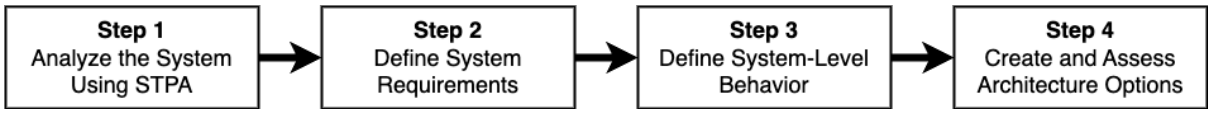
Step 1
Analyze the System Using STPA
Step 2
Define System Requirements
Step 3
Define System-Level Behavior
Step 4
Create and Assess Architecture Options

Figure 2. The STPA-Based Architecture Development Process (Poh, 2022)

The remainder of this paper demonstrates the STPA-based design process shown in Figure 2 by applying it to develop the pilot-automation architecture for an advanced rotorcraft concept to be flown in degraded visual environment (DVE) conditions such as rain, fog, or snow.

Case Study Results

Steps 1 & 2: STPA Analysis of Rotorcraft and Generation of Safety Requirements

STPA starts by defining the losses and system hazards (Leveson & Thomas, 2018). This case study focused on analyzing two main losses: (L-1) loss of life or injury and (L-2) damage to the aircraft. The system hazard that was analyzed was (H-1) violation of minimum separation with obstacles, other aircraft or terrain. Other losses and hazards are listed in (Poh, 2022).

Next, the control structure is created. The initial control structure for the rotorcraft is shown in Figure 3. Note that an abstract controller called the *Piloting Controller* represents any human pilots or automated flight controllers. By abstracting away any notion of the controller of the aircraft being human or automated, the system-level requirements needed to safely fly the aircraft can be generated first. More informed decisions can then be made later about what role a human pilot or automation should play in flying the aircraft.

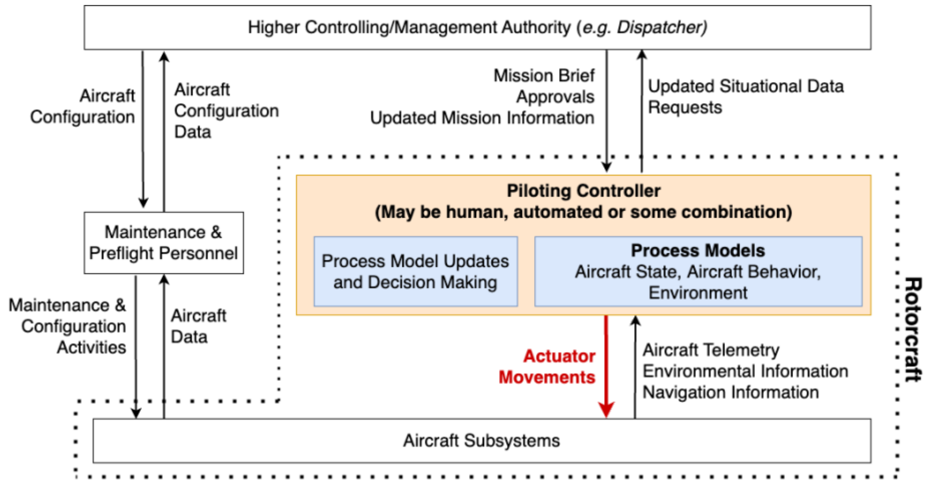
Higher Controlling/Management Authority (e.g. Dispatcher)
Aircraft Configuration
Aircraft Configuration Data
Mission Brief
Approvals
Updated Mission Information
Updated Situational Data
Requests
Maintenance & Preflight Personnel
Piloting Controller
(May be human, automated or some combination)
Process Model Updates and Decision Making
Process Models
Aircraft State, Aircraft Behavior, Environment
Maintenance & Configuration Activities
Aircraft Data
Actuator Movements
Aircraft Telemetry
Environmental Information
Navigation Information
Rotorcraft
Aircraft Subsystems

Figure 3. Rotorcraft Control Structure (Poh, 2022)

Using this control structure, the *Actuator Movements* control action shown in red in Figure 3 was analyzed to generate unsafe control actions (UCAs) and causal scenarios. Table 1 shows one example UCA, two possible causal scenarios, and the system requirements derived to prevent them. Additional examples can be found in (Poh, 2022).

Step 3: Defining Rotorcraft System-Level Behavior

The next step is to define the system-level behavior by identifying the responsibilities, process model parts, control actions, feedback, and timing requirements needed to meet the system requirements and adequately control the hazards. For autonomous systems, clearly defining tasks, task elements, and goals and expectations provides the rationale for each system design element and makes it easier to determine the effects of that design on human performance and safety. Table 2 shows an example of the system-level behavior defined for the responsibility that meets requirement Req-1 in Table 1.

Table 1. Example UCA, Causal Scenarios and Derived Requirements

UCA: Piloting Controller **provides** actuator movements that steer the aircraft toward another aircraft or object [H-1]

Causal Scenario	**System Requirement**
CS-1: Due to poor visibility conditions (e.g. fog), the piloting controller receives inaccurate sensor feedback that wrongly indicates no aircraft or objects nearby. The piloting controller thus wrongly decides to steer the aircraft toward an aircraft or object it does not realize is present.	**Req-1:** The aircraft system must be able to detect and track all objects and other aircraft in the environment under all DVE conditions.
CS-2: Wind gusts move the aircraft toward another aircraft or object so forcefully that, due to poor visibility conditions, the piloting controller does not react quickly enough or with sufficient force to avoid a collision.	**Req-2:** The aircraft system must respond quickly enough and with appropriate magnitude to prevent unintended aircraft movement.

Table 2. Example System-Level Behavior Definition

Responsibility	**Resp-1:** Detect and track all objects and other aircraft in the environment under all DVE conditions *[meets Req-1]*
Process Model Parts & Feedback	• Positions and speeds of other aircraft *(from aircraft transponders)* • Locations of ground obstacles *(from charts and sensor feedback)* • Current weather (e.g. temperature, wind speed) *(from sensor feedback and weather forecasts)*
Control Actions	Consolidated Airspace State *[used by Resp-3]*
Timing Requirements	<Defined based on how often the airspace state must be updated to identify a collision early enough to take action to prevent it>

These system-level behavior elements are closely related to aspects of information processing models (Proctor & Van Zandt, 2018) and can be used to inform the assignment of responsibilities to the human operator or automation. For example, the decision making or timing requirements can help determine the mental workload imposed on a human operator. Similarly, the process model parts can help determine if a responsibility imposes significant memory loads or high perception, comprehension or situational awareness requirements if it were assigned to a human operator. As a result, the criticality of required information (e.g. process model parts) in terms of system performance (e.g., latency, integrity) becomes evident.

Step 4: Creating and Assessing Rotorcraft Architecture Options

In this design process, architecture options are created by considering possible ways to assign the responsibilities (defined in the system-level behavior) to system controllers. In this case study, three architecture options were considered. Option 1 is a low-automation option that relies on manual flight. Option 2 is a medium-automation option where the human pilot and automation work together to fly the aircraft. Finally, Option 3 is a high-automation option where the human pilot primarily supervises the automation that flies the aircraft. Table 3 shows how four example responsibilities from (Poh, 2022) are assigned to a human pilot or an automated software-enabled controller (ASEC) in each architecture option. By defining expected human interactions in each architecture option like this, it is easier to assess an architecture's impact on human performance, especially in off-nominal situations. The four responsibilities are:

- **Resp-1:** Detect and track all objects and other aircraft in the environment under all DVE conditions at all times
- **Resp-2:** Ensure that a viable, collision-free flight path is always available
- **Resp-3:** Select a flight path that maintains separation with nearby objects or aircraft
- **Resp-4:** Respond quickly enough and with appropriate magnitude to execute the desired flight path and prevent unintended movement of the aircraft

Table 3. Assignment of Example Responsibilities in Each Architecture Option

Resp. ID	**(1) Low Automation**		**(2) Medium Automation**		**(3) High Automation**	
	Pilot	ASEC	Pilot	ASEC	Pilot	ASEC
Resp-1	●	●	●	●	●	●
Resp-2	●		●	●	●	●
Resp-3	●		●	●	●	●
Resp-4	●	●	●	●		●

Analyzing these architecture options using STPA identified three human factors-related tradeoffs. First, each successive architecture option is intended to reduce the human pilot's workload, but other factors can negate the potential workload savings. For example, although, option 2 shares more responsibilities between the pilot and ASEC than option 1, the number of process model parts maintained by the pilot is not reduced because they still need them to make their decisions. Maintaining these process model parts also ensures the human pilot does not experience an escalation of workload if the automation enters a degraded condition or behaves in an unsafe way (Copeland, 2019). Thus, the pilot's required level of situational awareness is not reduced despite sharing more responsibilities with the ASEC, and they must expend additional effort to coordinate with the ASEC to make safe decisions in flight. Similarly, option 3 further reduces the human pilot's workload by assigning to the ASEC the responsibility for controlling the aircraft quickly and accurately (Resp-4). However, this only eliminates a few mental model parts for the human pilot and further increases the effort required to coordinate with the ASEC.

The second human factors tradeoff is that regardless of how much automation is used, the system must always be designed to ensure that unsafe decision-making biases or heuristics do not lead to unsafe system behavior. At low levels of automation, the system design should assist the human pilot in avoiding unsafe biases or heuristics when making decisions or interpreting feedback. At higher levels of automation, the system still needs to be designed to avoid unsafe decision-making by the pilot because they can influence the ASEC's behavior. The system design must therefore include appropriate human-automation interactions to enable (1) operator insight (an accurate mental model of system behavior), (2) adequate operator oversight of the system states and behaviors, and 3) timely operator intervention if necessary (Copeland, 2019; Copeland et al., 2024).

The last human factors tradeoff is that as the level of automation increases, it becomes increasingly important to ensure the system is designed to enable good coordination between the human pilot and ASEC. Good coordination ensures that any process model parts shared between the human pilot and ASEC remain synchronized, and they act in a coordinated manner. It also ensures the human pilot and ASEC can understand each other's decision rationale to resolve conflicting decisions or enable the pilot to recover from unsafe behavior of the automation.

Using these tradeoffs, a system architect can make more informed decisions about which responsibilities *should* be assigned to the human pilot or automation and thus decide the role of the human pilot and how much automation to employ in the rotorcraft architecture.

Conclusion

This paper demonstrates how an STPA-based approach to architecture development enables earlier and more integrated consideration of human factors and safety during design. Three benefits of this new approach were demonstrated. First, STPA ensures human factors and safety considerations are included when generating system requirements. Second, defining system-level behavior provides human factors-relevant design information and identifies better information requirements. Finally, comparing architecture options highlights human factors-related tradeoffs and ensures the architecture includes adequate human-automation interactions.

References

Copeland, R. (2019). *An Analysis and Classification Process towards the Qualification of Autonomous Systems in Army Aviation*. Vertical Flight Society's 75th Annual Forum & Technology Display.

Copeland, R., Carter, H., & Mulder, J. (2024). *Information Management and Information Fusion: Human Engineering Approaches and Constructs (Revisited)* [US Army, DEVCOM AvMC, Redstone Arsenal, AL (Whitepaper)].

Flanigen, P., Copeland, R., Sarter, N., & Atkins, E. (2022). Current challenges and mitigations for airborne detection of vertical obstacles. *Proceedings for International Society for Optics and Photonics (SPIE) Defense and Commercial Sensing*.

Leveson, N. (2011). *Engineering a safer world: Systems thinking applied to safety*. MIT Press.

Leveson, N., & Thomas, J. P. (2018, March). *STPA Handbook.* psas.scripts.mit.edu/home/get_file.php?name=STPA_handbook.pdf

Parasuraman, R., Sheridan, T. B., & Wickens, C. D. (2000). A model for types and levels of human interaction with automation. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, *30*(3), 286–297.

Poh, J. (2022). *A Top-Down, Safety-Driven Approach to Architecture Development for Complex Systems* [Masters]. MIT.

Proctor, R. W., & Van Zandt, T. (2018). *Human factors in simple and complex systems* (3rd ed.). CRC Press.

SAE International. (2021). *J3016: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*.

Wickens, C. D. (Ed.). (2004). *An introduction to human factors engineering* (2nd ed.). Pearson/Prentice Hall.