



SURFACE VEHICLE RECOMMENDED PRACTICE

J3187™

FEB2022

Issued

2022-02

System Theoretic Process Analysis (STPA) Recommended Practices for Evaluations of Automotive Related Safety-Critical Systems

RATIONALE

Scope of this effort intends to provide both educational materials and recommended practices regarding how system theoretic process analysis (STPA) may be applied within a safety assessment process focusing on safety-critical content.

Purpose of this task force is to align industry (starting with, but not limited to, automotive/aerospace) best practices and translate them across industry regarding the implementation and use of STPA across human- and software-intensive systems (controls, human machine interactions (HMI), autonomous, etc.), and to explore focus areas suited for STPA use, or for supplementing other safety tools.

TABLE OF CONTENTS

1.	SCOPE.....	6
1.1	Purpose.....	6
2.	REFERENCES.....	6
2.1	Applicable Documents	6
2.1.1	SAE Publications.....	6
2.1.2	Other Publications.....	6
3.	DEFINITIONS	8
4.	ACRONYMS	10
5.	DOCUMENT ORGANIZATION.....	11
6.	BASIC STPA APPROACH AND LESSONS LEARNED.....	11
6.1	Introduction	11
6.2	STPA Method Overview.....	11
6.3	Use of Systems Engineering Perspective.....	13
6.4	STPA Scope Determination	16
6.5	Potential Loss ("Accident") Identification.....	17
6.6	Potential "Hazard" Identification.....	18
6.7	Defining System-Level Constraints (Safety Goals).....	19
6.8	Creating a Control Structure	20
6.9	Define Unsafe/Unwanted/Unexpected Control Actions (UCAs)	23
6.10	Define Casual Scenarios.....	26
6.11	Creation of Safety Requirements	30
6.12	Lessons Learned.....	32
6.13	Questions to Prepare for STPA	33

SAE Executive Standards Committee Rules provide that: "This report is published by SAE to advance the state of technical and engineering sciences. The use of this report is entirely voluntary, and its applicability and suitability for any particular use, including any patent infringement arising therefrom, is the sole responsibility of the user."

SAE reviews each technical report at least every five years at which time it may be revised, reaffirmed, stabilized, or cancelled. SAE invites your written comments and suggestions.

Copyright © 2022 SAE International

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of SAE.

TO PLACE A DOCUMENT ORDER: Tel: 877-606-7323 (inside USA and Canada)
Tel: +1 724-776-4970 (outside USA)
Fax: 724-776-0790
Email: CustomerService@sae.org
http://www.sae.org

SAE WEB ADDRESS:

Full document details: SAE J3187_202202

For more information on this standard, visit
https://www.sae.org/standards/content/J3187_202202/

Documents

Document Sharing Hub

6.14	Questions While Performing STPA.....	33
6.15	Lessons Learned for Incorporating STPA in Large Organizations	34
6.16	Japan Automotive Software Platform and Architecture (JASPAR) Lessons Learned	34
6.16.1	Control Structure Legend	35
6.16.2	Vehicle Level Control Structure	36
6.16.3	System-Level Control Structure	36
7.	SAFETY OF THE INTENDED FUNCTIONALITY (SOTIF) AND STPA	37
7.1	Mapping STPA to SOTIF Clauses	37
7.2	Unique Aspects of SOTIF for Specification and Testing Phase	38
7.3	Specification Phase: Using STPA for Requirements Identification.....	40
7.4	Identification of Relevant Control Actions	40
7.5	SOTIF Clause 6: SOTIF Hazard Identification and Risk Evaluation.....	41
7.6	SOTIF Clause 9: Definition of Verification and Validation Strategy	42
7.7	Testing Phase	42
7.8	Testing Phase: Using STPA for Scenario Generation	42
7.8.1	Scenery, Environment, and Dynamic Elements	43
7.8.2	Additional Context	44
7.8.3	Pass Criteria/Validation Targets	44
7.9	Case Study: Low-Speed Automated Driving System	45
7.9.1	STPA Step 1	46
7.9.2	STPA Step 2: Control Structure	46
7.9.3	Step 3: Unsafe Control Action.....	49
7.9.4	Step 4: Loss Scenarios	49
7.9.5	Scenario Generation Based on STPA Outputs.....	50
8.	HUMAN-MACHINE INTERACTION (HMI) AND STPA	52
8.1	Introduction	52
8.2	Human Extension Representation (HER)	53
8.3	STPA Lessons Learned for Human-Machine Interaction Evaluation	55
8.3.1	Understanding Expected System Operation, Behaviors, and Capabilities	55
8.3.2	Establishing System Scope	57
8.3.3	Considering Driver Actions	58
8.3.4	HMI UCA Considerations	59
8.4	Example: Shifting Vehicle to Park.....	59
8.4.1	Deciding When to Put Vehicle in Park	60
8.4.2	Conscious Specific Action to Initiate a Park Request.....	61
8.4.3	Assess Results of Request and Respond Accordingly.....	61
8.5	Dealing with Abstraction.....	62
8.6	Consulting with Driver Performance and Behavioral Experts Regarding Causal Scenarios.....	63
8.7	UCAs and Causal Scenarios with Human Extension Representation (HER).....	63
8.7.1	Feedback in Automotive HMI.....	66
8.7.2	Expansion with ADS.....	67
8.8	Generating Requirements and Traceability	67
8.8.1	Requirement Types.....	68
8.8.2	Connection to ISO 26262.....	69
8.9	Useful Definitions	70
8.9.1	Erroneous Interaction.....	70
8.9.2	Incidental Interaction	70
8.9.3	Purposeful Activation	70
9.	MODEL BASED SYSTEM ENGINEERING (MBSE) AND STPA	70
9.1	Modeling STPA	70
9.2	Overview of RAAML and OMG	71
9.3	Overview of Model-Based STPA	71
9.4	Example of Model Based STPA Applied to an Automotive System	73
9.4.1	Identifying Losses, Hazards, and Control Structure	73
9.4.2	Understanding Operational Contexts/Situations	76
9.4.3	Control Structure Representation	77

9.4.4	UCA Development	79
9.4.5	Loss Scenario Consideration	81
9.4.6	Expanding the Control Structure to Next Level of Detail	86
9.4.7	Transition to MBSE Environment	88
9.5	Summary of Benefits of Model Based STPA	88
10.	HIGH-LEVEL USE OF STPA WITHIN SAFETY PROCESSES AND STPA WITH OTHER SAFETY EVALUATION METHODS	89
10.1	Introduction	89
10.1.1	STPA Assistance in Established System Safety Processes and/or Standards	90
10.1.2	STPA Within a System Safety Process	99
10.2	Analysis Methodology Perspectives - Inductive, Deductive, and Exploratory	100
10.3	Comments on HAZOP and STPA	101
10.4	Interactions versus Interfaces	101
10.5	STPA in Human-Machine Interaction (HMI) Evaluations	102
10.6	STPA Compared to Other System Safety Analysis Methods	102
10.6.1	Useful Comparison Studies and Presentations	103
10.6.2	Illustrative Analogy for STPA and Other Methodologies	104
10.6.3	High-Level Methodology Comparison	104
10.7	Summary	106
10.7.1	STPA Value	106
10.7.2	STPA Application Perspective	106
10.7.3	STPA Effectiveness for Evaluating Human-Machine Interactions within Operating Scenarios	106
10.7.4	STPA and Emergent Properties and Behaviors	107
10.7.5	STPA within Existing Safety Evaluation Processes	107
11.	EXAMPLES	109
11.1	SOTIF Example (Okada and Christensen, 2017)	109
11.2	JASPAR STPA Example	147
11.2.1	Reasons for the Selection of the Target System	147
11.2.2	Main Functions of the Automated Driving System	147
12.	NOTES	161
12.1	Revision Indicator	161
Figure 1	Overview of the basic STPA method (Leveson and Thomas, 2018)	11
Figure 2	Initial allocation to major systems (Vernacchia, 2018)	14
Figure 3	Interactions between major systems (Vernacchia, 2018)	14
Figure 4	Rebalance and optimize initial concept (Vernacchia, 2018)	14
Figure 5	Selected concept/architecture (Vernacchia, 2018)	15
Figure 6	Multiple level balancing and optimization for continued concept/architecture improvement (Vernacchia, 2018)	15
Figure 7	STPA Handbook system level constraints (safety goals) (Leveson and Thomas, 2018)	20
Figure 8	High-level control structure	21
Figure 9	Abstraction progression example	22
Figure 10	Iterative cycle using STPA for system control structure updates	23
Figure 11	Guideword headings as shown in STPA Handbook (Leveson and Thomas, 2018)	24
Figure 12	Guideword table with additional helpful phrases (Ressler and Vernacchia, 2019)	24
Figure 13	Example of appropriate UCA format	25
Figure 14	Causal scenario graphic from STPA Handbook (Leveson and Thomas, 2018)	27
Figure 15	Compressed natural gas system: causal scenario "type" areas (Ressler and Vernacchia, 2018)	28
Figure 16	JASPAR lessons learned vehicle level control structure	36
Figure 17	JASPAR lessons learned system level control structure	37
Figure 18	Mapping Steps to SOTIF clauses	38
Figure 19	Scenario quadrants with awareness and safety as axes	39
Figure 20	Scenario quadrant with understanding and awareness as axes	39
Figure 21	ODD and ADS behavior with abstract functionality	40
Figure 22	High-level control architecture of an automated driving system	41
Figure 23	Scenery, environmental, and dynamic elements of a test scenario	43

Figure 24	Dynamic elements components	43
Figure 25	Control loop with process model beliefs	45
Figure 26	Low-speed automated driving system	45
Figure 27	Control structure for low-speed automated driving system case study	47
Figure 28	Sub-system under analysis for the case study	48
Figure 29	Driver as part of the control structure	53
Figure 30	Human extension of driver element in control structure	54
Figure 31	Detailed view of human extension representation (France, 2017)	54
Figure 32	Engineering “V” diagram highlighting concept of operations	55
Figure 33	System diagram (Vernacchia, 2016)	56
Figure 34	High-level system model representation	57
Figure 35	System scope outlined by dashed line	58
Figure 36	Substituting “driver” with human extension representation	60
Figure 37	Mental models	60
Figure 38	Control action selection	61
Figure 39	Mental model updates	61
Figure 40	Abstraction methods to manage complexity (Vernacchia, 2016)	62
Figure 41	Human extension representation example questions and answers	64
Figure 42	More HER example questions and prompts (France, 2017)	65
Figure 43	Tracing requirements to accidents (Ressler and Vernacchia, 2019)	68
Figure 44	RAAML overview package diagram (Object Management Group, 2021)	71
Figure 45	RAAML STPA method overview (Risk Analysis and Assessment Modeling Language (RAAML) libraries and profiles) (OMG, 2021)	72
Figure 46	Losses shown in the MBSE domain	74
Figure 47	Hazards shown in MBSE domain	75
Figure 48	Losses (accidents) and hazards in MBSE domain	76
Figure 49	Operational situations in the MBSE domain	77
Figure 50	Initial high-level control structure	78
Figure 51	High-level control structure in MBSE domain	79
Figure 52	UCAs associated with applying brakes when commanded by guidance system	80
Figure 53	UCAs in MBSE domain	80
Figure 54	MBSE domain guidewords linked to UCAs	80
Figure 55	MBSE domain UCAs linked through contexts to potential hazards	81
Figure 56	MBSE domain linkage of system element, functions, and UCAs	81
Figure 57	Scenarios and factors associated with UCAs and initial requirements	82
Figure 58	UCAs and loss scenarios in MBSE domain	82
Figure 59	Alternative view of loss scenarios and UCAs in MBSE domain (Object Management Group, 2020)	83
Figure 60	Another alternative view of control actions, unsafe control actions, and loss scenarios	83
Figure 61	Factors linked to loss scenarios in MBSE domain	84
Figure 62	Situations corresponding to Step 4 of STPA loss scenario assessment	85
Figure 63	Alternative view of loss scenario linkages	85
Figure 64	Scenarios and factors associated with UCAs and initial requirements	86
Figure 65	Expanded control structure with brake system sub-elements	87
Figure 66	Expanded control structure in MBSE domain	88
Figure 67	Systems engineering “V” model and STPA (Leveson and Thomas, 2018)	89
Figure 68	System engineering “V” model with STPA in a safety process (Vernacchia, 2018)	99
Figure 69	Different evaluation methodologies (Ressler and Vernacchia, 2019)	100
Figure 70	Causes and effects matrix (Ressler and Vernacchia, 2019)	101
Figure 71	Interactions of engine thrust (Vernacchia, 2016)	102
Figure 72	Hidden cube face	104
Table 1	System concept/architecture tasks involving early safety analysis methods (Vernacchia, 2018)	13
Table 2	JASPAR lessons learned control structure legend	35
Table 3	Functionality with ODD attributes and behavior competency (example specification)	41
Table 4	Set of UCAs for the case study	49
Table 5	Test scenarios based on STPA outputs	51
Table 5	Identified losses for intersection example	73
Table 6	Hazards associated with intersection example	74
Table 7	Operational contexts and links to potential hazards	76
Table 8	Standards comparison for requirements definition	92

Table 9	STPA high-level main steps	95
Table 10	STPA with ISO 26262, MIL-STD-882E, and ARP4761	96
Table 11	Attribute comparison of different analyses.....	105

1. SCOPE

Scope of this effort intends to provide both educational materials and recommended practices regarding how system theoretic process analysis (STPA) may be applied within a safety assessment process focusing on safety-critical content.

1.1 Purpose

Purpose of this task force is to align industry (starting with, but not limited to, automotive/aerospace) best practices and translate them across industry regarding the implementation and use of STPA across human- and software-intensive systems (controls, human machine interactions (HMI), autonomous, etc.), and to explore focus areas suited for STPA use, or for supplementing other safety tools.

2. REFERENCES

2.1 Applicable Documents

The following publications form a part of this specification to the extent specified herein. Unless otherwise indicated, the latest issue of SAE publications shall apply.

2.1.1 SAE Publications

Available from SAE International, 400 Commonwealth Drive, Warrendale, PA 15096-0001, Tel: 877-606-7323 (inside USA and Canada) or +1 724-776-4970 (outside USA), www.sae.org.

SAE J3016 Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles

Khastgir, S., Birrell, S., Dhadyalla, G., and Jennings, P., "The Science of Testing: An Automotive Perspective," SAE Technical Paper 2018-01-1070, 2018, <https://doi.org/10.4271/2018-01-1070>.

Okada, M. and Christensen, A. (2017, October 25). SAE Recommended Task Force - Group 4 Example.

Suo, D., Yako, S., Boesch, M., and Post, K., "Integrating STPA into ISO 26262 Process for Requirement Development," SAE Technical Paper 2017-01-0058, 2017, <https://doi.org/10.4271/2017-01-0058>.

2.1.2 Other Publications

Abrecht, B. (2016). A New Approach to Hazard Analysis for Naval Systems. MIT STAMP Workshop. Cambridge MA: MIT.

Crash Avoidance Metrics Partnership (CAMP) Automated Vehicle Research (AVR). (2016). Automated Vehicle Research for Enhanced Safety. Washington DC: U.S. Department of Transportation.

Fleming, C., Placke, M., and Leveson, N. (2013). Technical report: STPA analysis of NextGen interval management components: Ground interval management (GIM) and flight decn interval management (FIM).

France, M.E. (2017). Engineering for Humans: A New Extension to STPA. Cambridge MA: MIT - Masters Thesis.

France, M.E. (2017). ENGINEERING FOR HUMANS - Human-Automation Interaction in STPA. MIT STAMP/STPA Workshop (p. 18/50). Cambridge: MIT.

Georgiou, O.B. (2017). Haptic In-vehicle Gesture Controls. Proceedings of the 9th International Conference on Automotive User Interfaces and Interactive Vehicular Applications Adjunct (pp. 233-238).

Immersion Corporation. (2017). MAGNA and IPG Media Labs Haptic Ad Study. Immersion Corporation; <https://www.immersion.com/study-unveils-touchsense-ads-lead-to-brand-lift/>.

Kalra, N. and Paddock, S.M. (2016). Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? Transportation Research Part A: Policy and Practice.

Khastgir, S., Brewerton, S., Thomas, J., and Jennings, P. (2021). Systems Approach to Creating Test Scenarios for Automated Driving Systems. Reliability Engineering and System Safety.

Leveson, N. (2015). A systems approach to risk management through leading safety indicators. Reliability engineering and system safety.

Leveson, N. and Thomas, J. (2018). STPA Handbook. Cambridge MA: MIT.

Malloy, N.H. (2017). Integrating STAMP-Based Hazard Analysis with MIL-STD-882E Functional Hazard Analysis. MIT STAMP Workshop. Cambridge MA: MIT.

Martinez, P.D. (2017). Agency in Mid-air Interfaces. Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (pp. 2426-2439).

Martinez, R.S. (2015). A comparison of STPA and automotive FMECA. MIT STAMP Workshop. Cambridge MA: MIT.

Object Management Group. (2021). RAAML. Risk Analysis and Assessment Modeling Language (RAAML) Libraries and Profiles. Milford, MA, USA: OMG.

Object Management Group. (2020). RAAML Examples. Risk Analysis and Assessment Modeling Language (RAAML) Examples (Informative). Milford, MA, USA: Object Management Group.

OMG, O.M. (2021). Risk Analysis and Assessment Modeling Language (RAAML) Version 1.0. Milford, MA 01757 USA: OMG.

Pawlicki, T. (2018). Experiences with STPA in Radiation Therapy. MIT STAMP Workshop. Cambridge MA: MIT.

Pope, G. (2019). Risk Management Using Systemic Theoretic Process Analysis (STPA). Livermore, CA (United States): Lawrence Livermore National Lab (LLNL).

Reason, J. (1995). Understanding adverse events: human factors. Quality in Health Care, pp. 4:80-89.

Ressler, G.E. and Vernacchia, M.A. (2018). Designing in Safety Tutorial. International System Safety Society Conference Proceedings. Phoenix: ISSS.

Ressler, G.E. and Vernacchia, M.A. (2019). Helping Beginners Use System Safety Methodologies. International System Safety Conference ISSC37 (pp. 4, 131). Norfolk VA: ISSS - International System Safety Society.

Ribeiro, D.D. (2016). A Systems Approach to the Development of an Aircraft Smoke Control System. MIT STAMP. Cambridge MA: MIT.

Silvis-Cividjian, N., Verbakel, W., and Admiraal, M. (2018). Application of STPA in Radiation Therapy: a Preliminary Study. MIT STAMP Workshop. Cambridge MA: MIT.

Skedung, L.A. (2012). Feeling Small: Exploring the Tactile Perception Limits. Scientific Reports, 1-6.

Thomas, J. (2020). 8.4.1.7 When STPA Results Surprise You - An industry case study employing STPA, Fault Trees, FMEA, and HAZOP. MIT STAMP Virtual Workshop. Cambridge MA: MIT.

Thomas, J. and Gibson, M. (2020). Industry Trials to Evaluate STPA's Effectiveness and Practicality for Digital Control Systems. MIT STAMP Virtual Workshop. Cambridge MA: MIT.

Vernacchia, M.A. (2016). Defining Safety Requirements for Human-Machine Interactions. International System Safety Conference (p. 6). Orlando: ISSS.

Vernacchia, M.A. (2016). Internal General Motors System Safety Training. Milford MI: General Motors Company.

Vernacchia, M.A. (2017). System Concept Analysis Methods. Milford: GM Internal Document.

Vernacchia, M.A. (2018). Delivering Safety Through Early Analysis Methods. ISSC 36 Conference Proceedings (p. 13). Phoenix: ISSS.

Vernacchia, M.A. (2018). Integration of STPA into GM System Safety Process. MIT STAMP Workshop (p. 5). Cambridge MA: MIT.

Vernacchia, M.A. (2019). Integrating STPA into Large Organizations - Lessons Learned at General Motors. ISSC37 - International System Safety Conference. Norfolk VA: International System Safety Society.

Vincoli, J.W. (2005). Basic Guide to System Safety, Second Edition. John Wiley and Sons, Inc., Hoboken, NJ, USA.

Young, W. (2018). STPA in Cybersecurity. MIT STAMP/STPA Workshop. Cambridge: MIT.

3. DEFINITIONS

ACCIDENT: Unforeseen and unplanned event or circumstance resulting in injury, pain, harm, or loss. (Leveson, MIT)

ARCHITECTURE: Representation of the structure of the item or functions or systems or elements that allows identification of building blocks, their boundaries, and interfaces, and includes the allocation of functions to hardware and software elements. (ISO 26262)

CONCEPT OF OPERATIONS: A document produced early in the requirements definition process to describe what the system will do (not how it will do it) and why (rationale).

NOTE 1: Usually referred to as “ConOps.”

NOTE 2: It should also define any critical, top-level performance requirements or objectives (stated either qualitatively or quantitatively) and system rationale. (INCOSE Systems Engineering Handbook)

CONDITION: State of something, especially with regard to its appearance, quality, or working order. (ISSS working group)

CONTROL ACTION: A command or feedback item created by, or used by, a system element to perform its function(s).

NOTE: Functions “do things” and control actions are the commands/activities the functions drive and/or the feedback needed by the element to perform the function.

CONTROLS STRUCTURE: Hierarchical structures where each level imposes constraints on the activities of the level beneath them and accidents are viewed as the consequence of inadequate control of safety constraints.

NOTE: (<https://systemsthinkinglab.com/stamp/>); provides a method to document and graphically depiction of the functional design of the system (Leveson); systems represented as interrelated components kept in a state of dynamic equilibrium by feedback loops of information and control. (Leveson)

DEDUCTIVE ANALYSIS: Analysis approach starting out with a general statement, or hypothesis, and examines the possibilities to reach a specific, logical conclusion; the general to the specific. (<https://www.livescience.com/21569-deduction-vs-induction.html>)

ELEMENT: System or part of a system including components, hardware, software, hardware parts, and software units. (ISO 26262)

EXPLORATORY ANALYSIS: Analysis approach that considers multiple points of view and tries to anticipate all possible objections to, or flaws in, a particular position, with the goal of seeking the truth; used when cause and effects may not be known or not known in sufficient detail. (https://en.wikipedia.org/wiki/Exploratory_thought)

FAIL SAFE: Characteristic of a system whereby any malfunction affecting the system safety will cause the system to revert to a state that is known to be within acceptable risk parameters. (FAA System Safety Handbook)

FAILURE: Termination of the ability of an element to perform a function as required. (ISO 26262)

FAULT: Abnormal condition that can cause an element or an item to fail. (ISO 26262)

HARM: Physical injury or damage to the health of persons. (ISO 26262)

HAZARD: Potential source of harm caused by malfunctioning behavior of the item. (ISO 26262)

HAZARD ASSESSMENT: Process involved in determining the hazard level. (Leveson)

HUMAN FACTORS (ERGONOMICS): Applied science concerned with designing and arranging things people use so that the people and things interact most efficiently and safely. (Websters)

INDUCTIVE ANALYSIS: Analysis approach using Inductive reasoning makes broad generalizations from specific observations, the specific to the general. (<https://www.livescience.com/21569-deduction-vs-induction.html>)

INTERACTION: System state affects or influences one or more vehicle systems, or system elements, even if they may not be connected or communicating with each other.

INTERFACE: Occurrence of direct data/info transfer and/or communication between two systems or system elements within a system.

LOSS: Damage, trouble, disadvantage, deprivation, etc., caused by losing something. (<http://www.yourdictionary.com/loss>)

MALFUNCTION: A failure to function in a normal or satisfactory manner. (Oxford Languages)

MISHAP: Unfortunate accident. (ISSS Working Group)

REQUIREMENTS: Statements describing essential, necessary, or desired attributes. (FAA System Safety Handbook)

RISK: The combination of the probability of occurrence of harm and the severity of that harm. (ISO 26262)

RISK—ACCEPTABLE: Risk that is understood and agreed to by the program/project, governing authority, mission directorate, and other customer(s) such that no further specific mitigating action is required. (NASA Systems Engineering Handbook)

RISK—RESIDUAL: Risk remaining after the deployment of safety measures. (ISO 26262)

RISK—UNREASONABLE: Risk judged to be unacceptable in a certain context according to valid societal moral concepts. (ISO 26262)

RISK ASSESSMENT: Process of determining the risk level (quantifying risk). (Leveson)

RISK LEVEL: Function of the hazard level combined with (1) the likelihood of the hazard leading to an accident, and (2) hazard exposure or duration. (Leveson)

SAFETY: Absence of unreasonable risk (1.136). (ISO 26262)

SUB-SYSTEM: An element of a system that, in itself, may constitute a system. (FAA System Safety Handbook)

SYSTEM: Collection of elements which, working together, produce the desired result not achievable by the elements alone.

SYSTEM SAFETY ENGINEERING: Disciplined, systematic, comprehensive engineering effort systems, within systems engineering, to identify safety-related risks and eliminate or control them.

SYSTEMS ENGINEERING: Engineering discipline whose responsibility is creating and executing an interdisciplinary process to ensure that the customer and stakeholder's needs are satisfied in a high quality, trustworthy, cost-efficient, and schedule compliant manner throughout a system's entire life cycle. (INCOSE)

TRADE OFF: Decision making actions that select from various requirements and alternative solutions on the basis of net benefits to the stakeholders. (INCOSE)

TRADE STUDY: A means of evaluating system designs by devising alternative means to meet functional requirements, evaluating these alternatives in terms of the measures of effectiveness and system cost, ranking the alternatives according to appropriate selection criteria, dropping less promising alternatives, and proceeding to the next level of resolution, if needed. (NASA Systems Engineering Handbook)

UNSAFE CONTROL ACTION: Control action that, in a particular context and worst-case environment, will lead to a hazard.

VERIFICATION: Confirmation that work products properly reflect the requirements specified for them ("you built it right"). (NASA Systems Engineering Handbook)

VALIDATION: Confirmation that the product fulfills its intended use in all of the environments that the product will be used in ("you built the right thing"). (NASA Systems Engineering Handbook)

4. ACRONYMS

DFMEA	Design Failure Modes and Effects Analysis
FIA	Functional Interface Analysis
FMEA	Failure Modes and Effects Analysis
FTA	Fault Tree Analysis
HMI	Human-Machine Interaction
INCOSE	International Council on Systems Engineering
ISO	International Organization for Standardization
JASPAR	Japan Automotive Software Platform And Architecture
MBSE	Model-Based Systems Engineering
NASA	National Aeronautics and Space Administration
OMG	Object Management Group
RAAML	Risk Analysis and Assessment Modeling Language
SOTIF	Safety of the Intended Functionality
STPA	System Theoretic Process Analysis
UCA	Unsafe/Undesired Control Action

5. DOCUMENT ORGANIZATION

This document is organized as follows:

- Basic STPA approach, recommended practices, lessons learned
- SOTIF and STPA
- HMI and STPA
- MBSE and STPA
- High-level use of STPA within safety process and STPA with other safety evaluation methods
- Examples (aerospace, automotive, automotive HMI, MBSE, SOTIF)

6. BASIC STPA APPROACH AND LESSONS LEARNED

6.1 Introduction

This section describes attributes of a basic system theoretic process analysis (STPA) approach. The content in this section comes from practitioner experience, and at times references, and excerpts from the STPA Handbook (Leveson and Thomas, 2018).

NOTE: This section does not intend to teach the STPA process, as other resources such as the STPA Handbook accomplish that goal. Please refer to these other resources for specific STPA questions. See Section 2 for additional resources.

6.2 STPA Method Overview

There are four steps in a basic STPA system safety evaluation. This section outlines these steps and provides recommendations to effectively execute these steps. The steps in a basic STPA are shown in [Figure 1](#) (excerpt from STPA Handbook), along with a graphical representation of these steps.

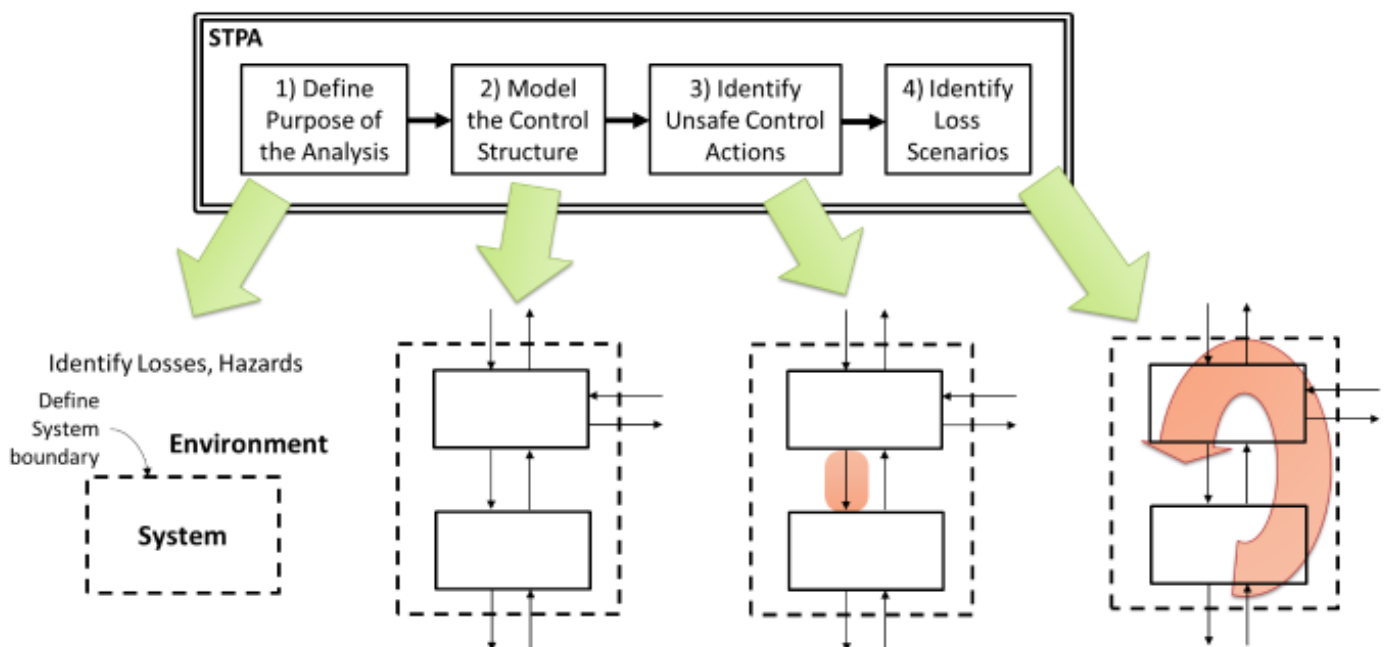


Figure 1 - Overview of the basic STPA method (Leveson and Thomas, 2018)

Used with permission from Leveson and Thomas. All other rights reserved.

The first step is to define the purpose and scope of the STPA evaluation effort. Accidents (losses) and safety goals associated with automotive applications are defined along with potential hazardous states (hazards) that may result from system element misbehaviors.

STPA can be used as a standalone process, or it can be used alongside existing frameworks. When used in an ISO 26262 framework, the STPA accidents and hazards are associated with potential harm done to human operators, vehicle occupants, and surrounding pedestrians. (SAE J2980 is a good reference with examples of automotive hazard analysis.) When used in other frameworks, the STPA accidents and hazards can be linked to mishaps, hazards, or similar concepts.

High-level system constraints (safety goals) are developed by reformatting the hazards into constraints that will address the specific hazard. When used in an ISO 26262 framework, the STPA high-level system constraints can be listed as “safety goals,” which provides an approach to link safety requirements to safety goals and then to hazards¹. When used in other frameworks, the STPA high-level system constraints can be linked to safety objectives, high-level safety requirements, or similar concepts.

Building a hierarchical “control structure” is the second step. A control structure captures the functional interactions between system elements using a series of command and feedback loops. It is advantageous to model the system using a high-level control structure at an abstract level and then refine it to more detailed levels as the evaluation progresses. The control structure should not include implementation details but rather logical elements representing software controllers, actuators, sensors, and processes that the system controls. A control structure need not be limited to electromechanical elements of the system but may extend beyond the technical system to include elements such as human operators, maintenance personnel, a control room or fleet operations center, and interactions with the external environment.

The third step is to identify unsafe or undesired control actions. This can involve examining functional responsibilities (functions) assigned to the various control structure elements. Undesired or unsafe control actions (UCAs) associated with each action that achieves a function are defined using a standard syntax and then linked to the hazards defined in the first step. A UCA may lead to more than one hazard, and hazards may result from multiple UCAs (many-to-many relationship). It is worth noting that developing UCAs may help identify additional hazards not identified in the first step.

STPA is a systematic process to reduce and eliminate open-ended mental processes that result in gaps and oversights. Therefore, this step should avoid using non-systematic methods such as unstructured brainstorming.

The final step identifies the circumstances, conditions, or reasons (causal scenarios/causal faults) why UCAs might occur in the system. These scenarios are created to explain:

1. How unsafe or undesired control action leads to a loss event.
2. How incorrect feedback, inadequate requirements, design errors, component failures, and other factors can independently or collectively contribute to unsafe or undesired control actions ultimately lead to losses.
3. How safe (or desired) control actions, including hazard mitigation actions, might be required but are not commanded, are commanded improperly, or executed improperly.

More than one causal scenario can exist for each unsafe or undesired control action.

Causal scenarios create additional technical requirements, drive the architecture, and highlight conflicting requirements that need to be resolved before the design can proceed.

An STPA analysis can provide recommendations and new design decisions if applied during early development. It can also:

- Evaluate design decisions and highlight gaps when applied to existing concepts.
- Identify functional and behavioral requirements.
- Define human procedures and training requirements.

¹ ISO 26262 defines a “safety goal” as a top-level safety requirement from the hazard analysis and risk assessment (HARA) at the vehicle level.

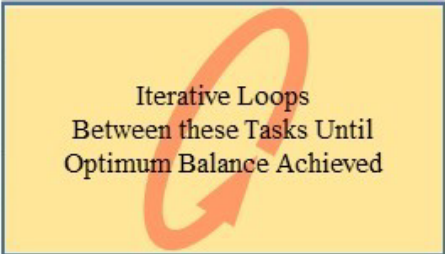
- Define test cases and create test plans.
- Develop leading indicators of risk.

6.3 Use of Systems Engineering Perspective

Effective safety analysis methods provide value through the entire engineering design, development, testing, and manufacturing phases of a system's implementation process. These analysis methods significantly impact system content in the early phases of a program as major expenditures for critical component procurements and manufacturing tooling have yet to be committed. There is much less cost associated with design changes at this point in the process than there are with changes that occur later in the process (e.g., it is much cheaper to move lines on paper than it is to move/change actual physical tooling lines in a manufacturing environment) (Vernacchia, 2018).

Early in the design process, exploration of various design concepts is conducted to determine the optimized balance between system imperatives such as cost, timing, mass, performance, and safety "acceptable risk" requirements. This "system architecture balance" phase typically contains tasks such as the ones shown here in [Table 1](#), with some tasks having an iterative nature between them. Of course, if an optimal balance cannot be achieved, the iterative nature would be extended to early tasks for modification of stakeholder expectations and initial requirements (Vernacchia, 2018).

Table 1 - System concept/architecture tasks involving early safety analysis methods (Vernacchia, 2018)
Used with permission from Vernacchia. All other rights reserved.

System Concept / Architecture Tasks	Safety Analysis Methods Areas
Stakeholder Expectation Determination	
Initial Requirements Identification	
Architecture Development / Modification	 Iterative Loops Between these Tasks Until Optimum Balance Achieved
Balance, Trade Studies, Optimization Activities Leading to Design Selection within Functional Domain(s)	
Analysis Verification (Functional, Performance, and Safety)	
Program Validation Against Stakeholder Expectations	
Concept / Architecture Selection	

An underlying assumption in [Table 1](#) is that system safety is an integrated part of the system engineering process to be optimally effective. The iterative loop shown in [Table 1](#) may be further illustrated by a systems engineering "cascading conic" depiction shown in [Figures 2, 3, 4, 5, and 6](#). [Figure 2](#) shows the initial allocation of system imperatives to major systems. [Figure 3](#) shows the interactions between these major systems during the balance, tradeoff, and optimization steps. [Figure 4](#) shows the rebalance and optimization at the top system level leading to an initial concept/architecture, and [Figure 5](#) depicts the concept/architecture selection.

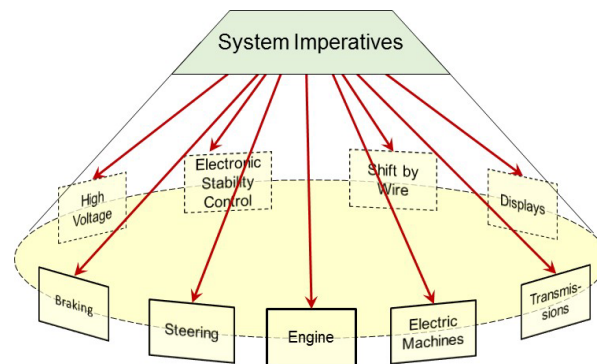


Figure 2 - Initial allocation to major systems (Vernacchia, 2018)
Used with permission from Vernacchia. All other rights reserved.

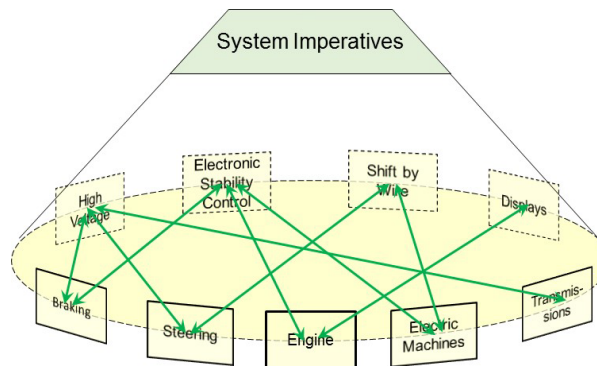


Figure 3 - Interactions between major systems (Vernacchia, 2018)
Used with permission from Vernacchia. All other rights reserved.

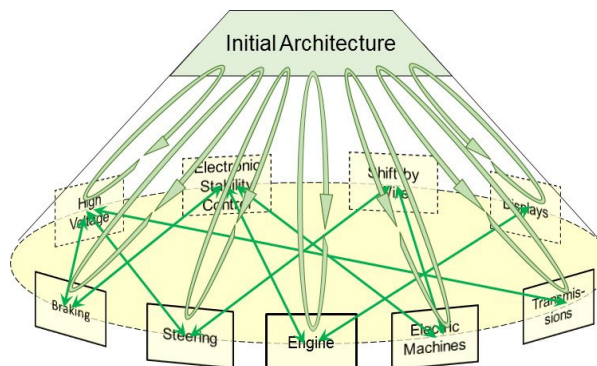


Figure 4 - Rebalance and optimize initial concept (Vernacchia, 2018)
Used with permission from Vernacchia. All other rights reserved.

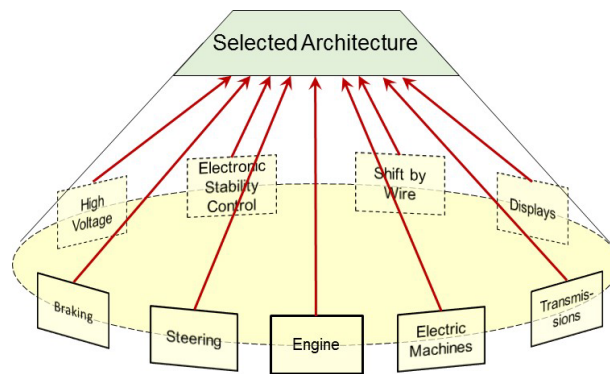


Figure 5 - Selected concept/architecture (Vernacchia, 2018)
Used with permission from Vernacchia. All other rights reserved.

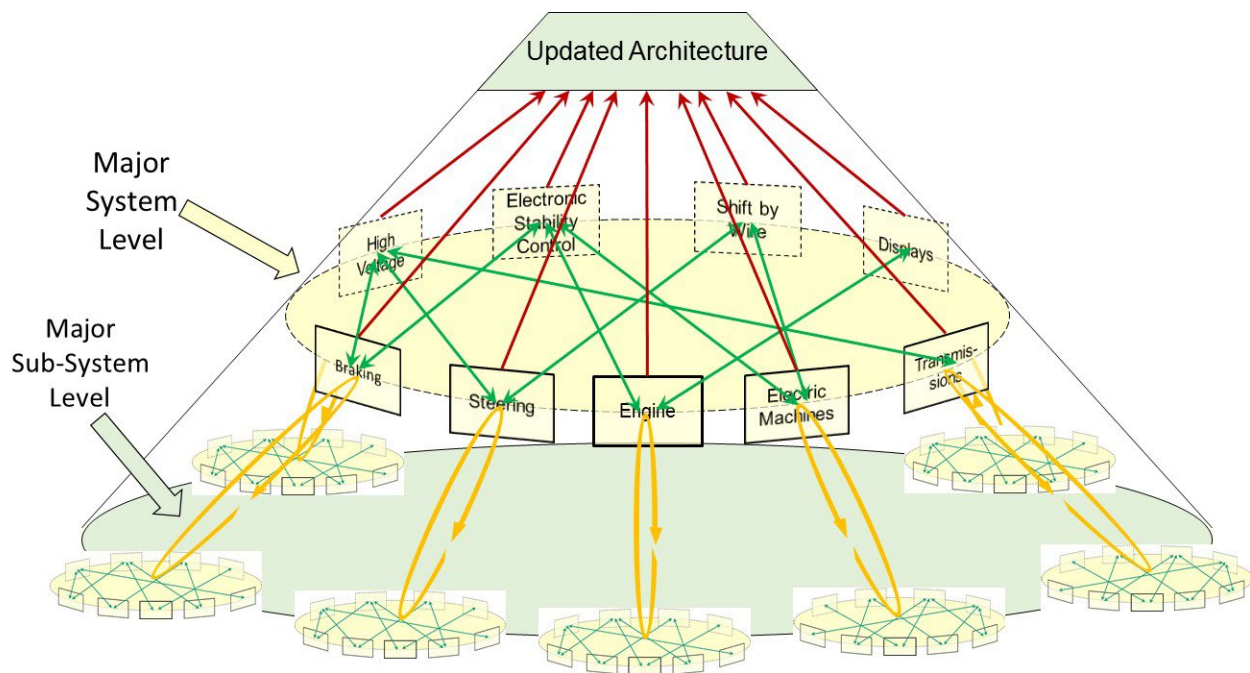


Figure 6 - Multiple level balancing and optimization for continued concept/architecture improvement (Vernacchia, 2018)
Used with permission from Vernacchia. All other rights reserved.

[Figure 6](#) illustrates the allocation to the major subsystems of each major system where the balance, tradeoff, and optimization process repeats at this lower level, and updates are transposed back up to the top level. Two levels of evaluation tend to be sufficient for initial safety task activities. Deeper levels of the engineering process are very well served by familiar evaluation methodologies such as DFMEAs, FTAs, and FIAs.

Within this systems engineering process framework, safety tasks performed early in the design process support the balance and optimization tasks by identifying what hazardous conditions could arise due to system element misbehavior and assessing the risk associated with such hazards. This effort defines the required system content and the detection and mitigation strategies necessary to eliminate or manage the assigned risk to acceptable levels. To be effective, these detection and mitigation strategies must be part of the balancing and optimization of the system.

A key asset for this top-down, iterative, and interactive process is a “chief system architect” who understands the technical nature of the major system interactions and should have the expertise to broker tradeoffs and compromises between functional areas. If no such person is available, it is useful to engage “system architects” from the different functional domains and engage them in the “roll-down” and “roll-up” activities involving system imperative and high-level constraints.

6.4 STPA Scope Determination

The first step is to determine the system scope under consideration and its hierarchal nature. The scope of the STPA analysis may change when going from the top hierarchal controller into a deep nested lower level controller, so the previous systems engineering-based section is an effective method to determine system content and element functions. Additional key questions found effective to consider when defining the system are:

- Define the purpose of the analysis, the strategy/focus of the analysis. (For example, is this a safety analysis, security analysis, systems analysis, or some combination of the three? This purpose should be identified early on as it will lead to the identification of losses to avoid as mentioned in the next section.)
- Is safety-relevant security included in the analysis? Security's impact on safety needs to be considered from the beginning. Correlating security-related UCAs and scenarios with the system hazards and losses identifies the causal factors that have priority (Young, 2018).
- List all elements at the highest abstract level.
- STPA should be applied initially at the highest abstract levels and then at more defined lower levels.
- The STPA control diagram is revisited throughout the analysis process, serving as a reference for the STPA analysis and guiding discussions with design and test engineering teams.
- For these reasons, it is important to balance the level of abstraction and detail such that all aspects of the system can be analyzed across diverse teams.
- A high level of abstraction is expected at the early definition stage, with details added as the project design progresses.
- Ask these upfront questions:
 - A system to do {What = Purpose}
 - By means of. {How = Method}
 - In order to contribute to {Why = Goals}
 - While {Constraints, restraints, ODD, etc.}
 - System actors {Who or what is interacting with the system?}
 - Delivered by {When = Milestone deliveries}
- The key part of this is capturing system assumptions. If system assumptions are not detailed early in the system scope and boundary, it will become apparent when trying to understand causal analysis and process model flaws. So, capture all assumptions of system behavior early on. Examples: "This analysis will only focus on automatic transmissions," or "This analysis is assumed to have Level 3 autonomy only."

Additional items to consider while determining scope and boundary are:

- List assumptions as to the basis for focused analysis.
- Think of ways to frame the problem.
- Avoid considering only the items we control, but intentionally look at all aspects with a specific interest in items that are not controllable (or perceived not to be controllable).
- Must ensure that we are defining and solving the right (engineering) problem.

- Group list of elements but avoid detailing lower levels and instead focus on top-level interactions of an element. It is okay to show some dependencies, but it is better to focus on top-level understanding.
- Consider not just communication but process interfaces as well (e.g., high voltage for electric powertrain).
- Identifies the system boundaries as well as defining the components (subsystems) inside the system boundaries targeted by STPA.
- Definitions of the components inside the system boundaries to the granularity of the control structure.

The next step is to identify potential accidents (losses) that may occur if certain hazardous states and specific circumstances are present.

6.5 Potential Loss (“Accident”) Identification

The next step is to identify potential accidents (e.g., “loss,” “harm”) that may occur if certain hazardous states and specific circumstances are present. In STPA, a “loss” can occur due to an accident. ISO 26262 uses “harm” as an equivalent for an accident. This document will use each of the three words—accident, loss, and harm—as equivalent terms.

STPA can be used to target any loss that is unacceptable to stakeholders. If more than one loss is included, they can be ranked and prioritized. Because every STPA result will be traceable to one or more losses, the analysis results can be easily ranked and prioritized based on the losses to which they refer.

Before any analysis begins, the stakeholders must identify the losses on which they want the analysis to focus. The losses to be considered may be defined by management, by government regulations, or by customers. A general approach to identifying losses may involve:

1. Identify the stakeholders, e.g., users, producers, customers, operators, etc.
2. Stakeholders identify their “stake” in the system. What do they value? For example, human life, a fleet of automobiles, electrical power generation, transportation, etc. What are their goals? For example, to maintain a fleet of automobiles, to provide transportation, to provide medical treatment, to provide electrical power generation, etc.
3. Translate each value or goal into a loss, e.g., loss of life, loss of vehicle functions, loss of electrical power generation, loss of transportation, etc.

Automotive application accidents will typically fall into one of the following categories:

- Vehicle collides with another vehicle.
- Vehicle collides with pedestrian(s).
- Vehicle occupant injury.
- Vehicle collides with infrastructure.

In addition to these categories, accidents (i.e., harm) result from other occurrences such as a fire or an electrical shock that may occur without a vehicle collision.

Depending on the purpose and scope of the STPA analysis, the evaluation of accidents (losses) can expand beyond safety-related incidents. Accidents can include various other types of losses, such as performance, financial, quality, risks, and threats. Accidents do not have to be limited to safety-related losses and can and should include other important aspects of the design as appropriate. A hazard need not always lead to a loss but may lead to a loss under certain conditions. For example, unintended braking may be a hazard for automated emergency braking (AEB). It may only lead to a loss if there is a following vehicle.

Additional key aspects to consider when identifying accidents and losses are:

- Use top-level strategy and priorities to focus on consequences directly related to the strategy, purpose, and definition of the system being analyzed.
- Verbs may be used to help identify system accidents and hazards:
 - AV colliding with mobile object.
 - AV injuring passengers without a collision.
- If security is included in the analysis, distinguish between accidents and losses, with losses defined as non-safety-related (loss of reputation, IP, privacy, or breach of customer personal information). In response to this note, the term accidents/losses are not defined and used separately in this document. The term and definition of accidents in this scope align with the automotive safety standard ISO 26262. Losses are a result of an accident, and losses can be safety-related or non-safety-related. Non-safety-related losses can be related to quality, performance, security, and financial.
- For traceability, use unique labels for each accident and loss (security loss not impacting safety).

Next is potential hazardous situations (system-level hazards) that may lead to accidents in the presence of specific circumstances need to be determined.

6.6 Potential “Hazard” Identification

Hazards should first be identified at the system level, i.e., associated with the high-level control structure representation. (More hazards may be defined as more detailed control structures are evaluated.) High-level control structure hazards for vehicle applications (e.g., road vehicles, aerial vehicles, space vehicles, etc.) might include vehicle exceeds safe operating envelope, vehicle does not maintain safe distance from nearby objects, or vehicle enters dangerous area/region. These hazards may lead to multiple accidents and losses. As such, the STPA process links each hazard to the resulting accidents and losses by typically documenting the accident/loss identifier in brackets after the hazard.²

The following criteria from the STPA Handbook (Leveson and Thomas, 2018) is useful for defining hazards:

- Hazards are system states or conditions (not component-level causes or environmental states).
- Hazards lead to an accident (or loss) in some worst-case environment.
- Hazards must describe states or conditions to be prevented.

The STPA Handbook goes on to outline some common mistakes when identifying hazards:

- Confusing hazards with causes of hazards.
- Too many hazards containing unnecessary detail.
- Ambiguous or recursive wording.

² In ISO 26262, “accidents” are not specifically linked to hazards. ISO 26262 investigates when hazardous situations may lead to “harm” under certain operating conditions or modes. An accident is an example of harm. Even with this distinction, ISO 26262 links hazards to an accident (harm) by hazard analysis and risk assessment (HARA). A system item is evaluated without internal safety mechanisms for various operational situations and operating modes. An operational situation or mode may define a hazardous event, which, unmitigated, may lead to a loss.

ISO 26262 provides one way to assess the impact of a potential loss using three parameters: severity, exposure, and controllability. Each of these parameters has multiple levels of degree and then can be combined to define an automotive safety integrity level (ASIL) rating. The five ISO 26262 classifications are (starting with most significant) ASIL D, ASIL C, ASIL B, ASIL A, and QM. QM (quality management) is the least significant classification and allows a robust engineering process to address system safety concerns.

- Confusing hazards with failures.
- Key pitfalls: If you use words like “causing” or “resulting in,” you are likely combining a hazard and a potential cause. See the STPA Handbook for examples.

It may be helpful to list out a few possible mechanisms for developing the initial hazard list before entering the UCAs (e.g., field data). The [link here](#) shows a high-level HAZOP to help find this initial list.

System-level hazards will be a list that will evolve as the STPA process is implemented, especially as the control diagram(s) are refined. Start with the list of elements created for a top-level understanding of the system. Use accidents that were identified earlier to identify related hazards, using verbs, avoid compound sentences:

- AV breaches the minimum safe distance of a forward mobile object.
- AV exposes passengers to unhealthy g-forces.

If security is included in the analysis, distinguish between hazards with the potential cause of an accident and losses that don't impact safety. Listing both adds clarity to the boundary of the analysis. As the analysis proceeds, it is possible that elements could move to the other list. It is important to list beyond the analysis boundary to leave room for the iterative nature of STPA.

For traceability, use unique labels for each hazard and loss.

6.7 Defining System-Level Constraints (Safety Goals)

The definition of STPA system-level constraints follows the hazard identification step in the STPA process. In this document, the term “safety goals” will be used.³

STPA offers a simple way to identify these system-level constraints by taking the hazard description and inverting the condition. A suggested format is provided here.

Hazard Listing:

Hazard Identifier -

System of Interest -

Unwanted/Unsafe Condition -

Association to Accidents/losses

System Level Constraint (SLC):

SLC Identifier -

System of Interest -

Constraint to Implement -

Hazards Prevented

³ ISO 26262 defines a “safety goal” as a top-level safety requirement resulting from hazard analysis and risk assessment (HARA). Non-safety-related losses, top-level security goals, and general system goals can also be created to link to associated risks and threats.

Using this format, two examples are presented here.

Hazard-1: Vehicle experiences an unwanted acceleration causing the driver to lose control (may lead to an accident where a collision with another vehicle or injury to occupant(s) or pedestrian(s)).

Safety Goal-1: Vehicle shall not experience an unwanted acceleration to a level that causes a driver to lose control where collision or injury could occur.

Hazard-2: High voltage battery causes electric shock (which may lead to an accident where humans are injured).

Safety Goal-2: High voltage system shall not allow an electric shock to injure humans.

These constraints are high level and, as such, are considered “safety goals.” Such safety goals will be assigned more detailed safety requirements as the system design develops. Having such safety goals enables more understandable traceability of requirements to hazards to accidents (losses).

Once the system-level hazards are identified, it is straightforward to identify system-level constraints that must be enforced: simply invert the condition.

<Hazard> = <System> & <Unsafe Condition> & <Link to Losses>

<Safety Constraint> = <System> & <Condition to Enforce> & <Link to Hazards>

H-1: Aircraft violate minimum separation standards [L-1, L-2, L-4, L-5]

SC-1: Aircraft must satisfy minimum separation standards from other aircraft and objects [H-1]

H-2: Aircraft airframe integrity is lost [L-1, L-2, L-4, L-5]

SC-2: Aircraft airframe integrity must be maintained under worst-case conditions [H-2]

Each constraint can be traceable to one or more hazards, and each hazard is traceable to one or more losses. In general, the traceability need not be one-to-one; a single constraint might be used to prevent more than one hazard, multiple constraints may be related to a single hazard, and each hazard could lead to one or more losses.

Figure 7 - STPA Handbook system level constraints (safety goals) (Leveson and Thomas, 2018)

Used with permission from Leveson and Thomas. All other rights reserved.

6.8 Creating a Control Structure

A control structure is a representation of the “control” hierarchy of the system. The controls hierarchy of a control structure typically starts out using a three-level structure showing the “system” being controlled at the lowest level, the “controller” driving that system at the mid-level, and the ultimate element commanding the controller at the highest level. An example would be the operator at the top of the hierarchy, a controller/computer interacting in the middle, and the system being controlled at the bottom. [Figure 8](#) shows such a control structure example.

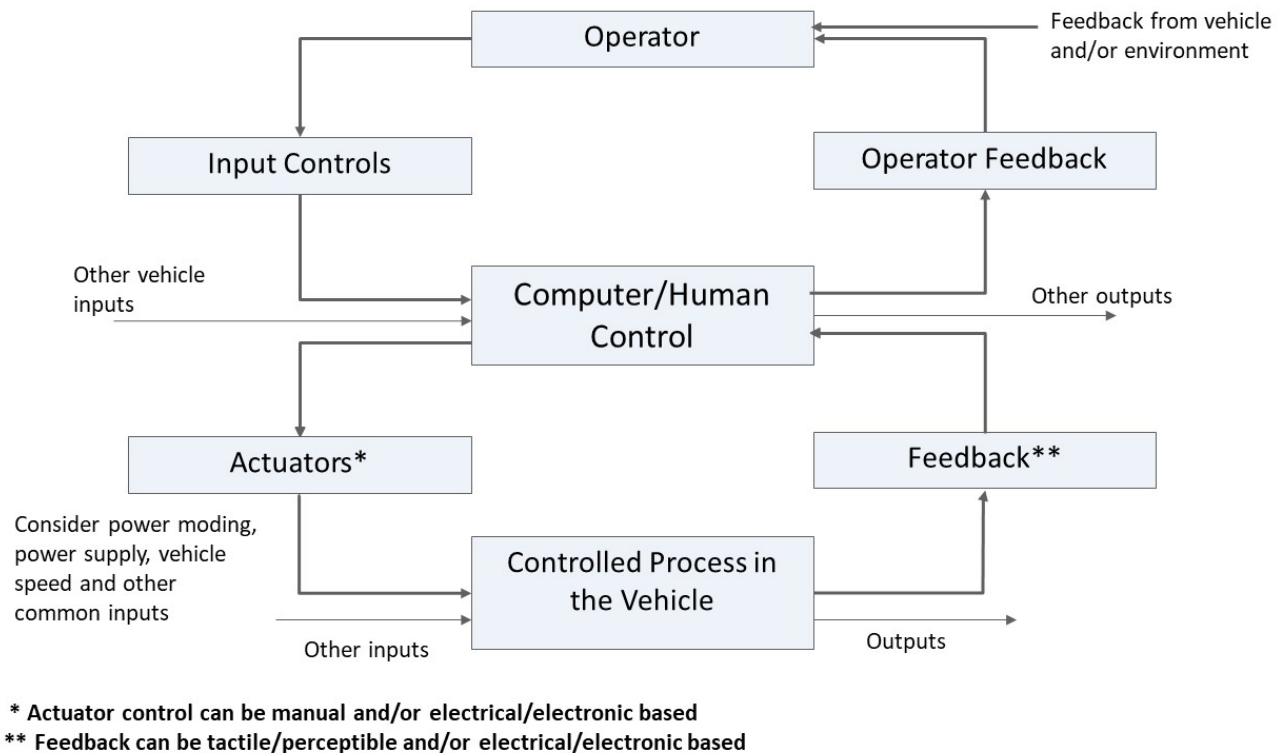


Figure 8 - High-level control structure

Vertical hierarchy denotes elements that are shown “higher” (towards the top) in the control structure have higher precedence of decision-making responsibilities. As such, the hierarchy indicates control authority between the system elements. Therefore, an element has control over the elements below it and is controlled by elements immediately above.

In addition to hierarchy, [Figure 8](#) also illustrates the control loops between system elements represented by the arrows. STPA notation convention uses downward arrows to convey control actions or commands and uses upward arrows to convey feedback.

The control structure representation of a system should begin at the highest level possible. This enables the use of abstraction to manage complexity. A dictionary definition of abstraction is “the process of taking away or removing characteristics from something to reduce it to a set of essential characteristics.” This approach allows the STPA practitioner to develop multi-level control structures with each lower level, illustrating the next level of complexity in the system. [Figure 9](#) shows a progression of abstraction; for example, compressed natural gas (CNG) propulsion systems.

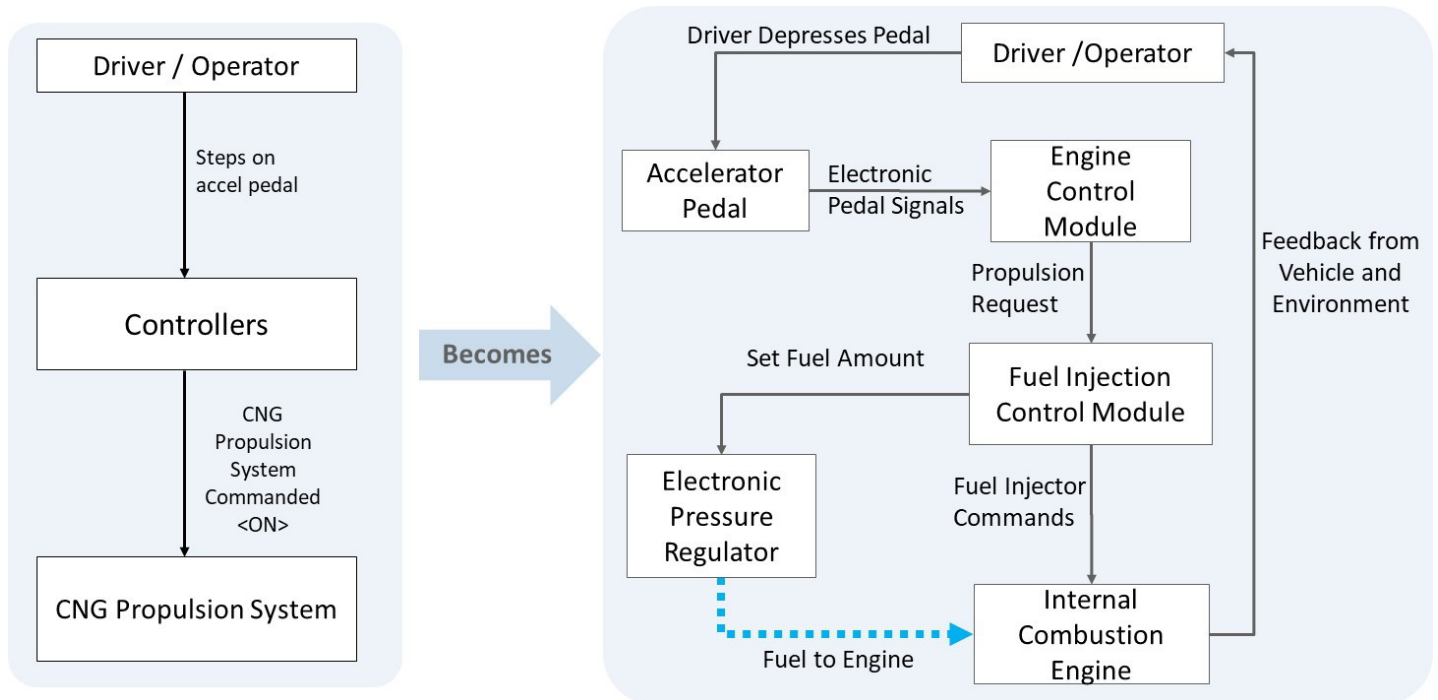


Figure 9 - Abstraction progression example

Once the control structure abstraction level is decided, the next step would be to assign functions to the control structure elements. These functions (aka “responsibilities”) may be achieved using command actions from elements, feedback activities from elements, and/or extensions of the system-level constraints. Each element should understand what it must do to be sure the safety goals are achieved. At times, these functions may also describe what the element is to do to enable the system to provide safe behaviors.

Abstraction is an STPA recommended approach to manage the complexity of the analysis. It allows higher-level representations of UCAs so that the evaluation will not bog down in numerous details too early in the process. This enables a practitioner to manage the complexity by deferring the detailed level analysis to later portions of the evaluation. In addition, it enables the identification of high-level requirements that would manage lower-level detailed requirements. In the end, the final requirements seem to be the same regardless of abstraction. Still, the non-abstraction approach has a risk of the evaluation collapsing as the effort required to complete the analysis appears too high.

Abstraction is also useful where numerous causal scenarios drive many requirements. A practitioner may observe that the same requirement addresses many causal scenarios.

Each analysis in a determined level of abstraction will link several activities. Evaluations of control structure elements lead to the identification of UCAs, which then lead to the determination of causal scenarios, which finally generate safety requirements or constraints. These safety requirements lead to design decisions with effects over the architecture, so functions/control actions are allocated to human/HW/SW. This leads to the construction of a more detailed functional control structure that leads to new UCAs, etc., as shown in [Figure 10](#).

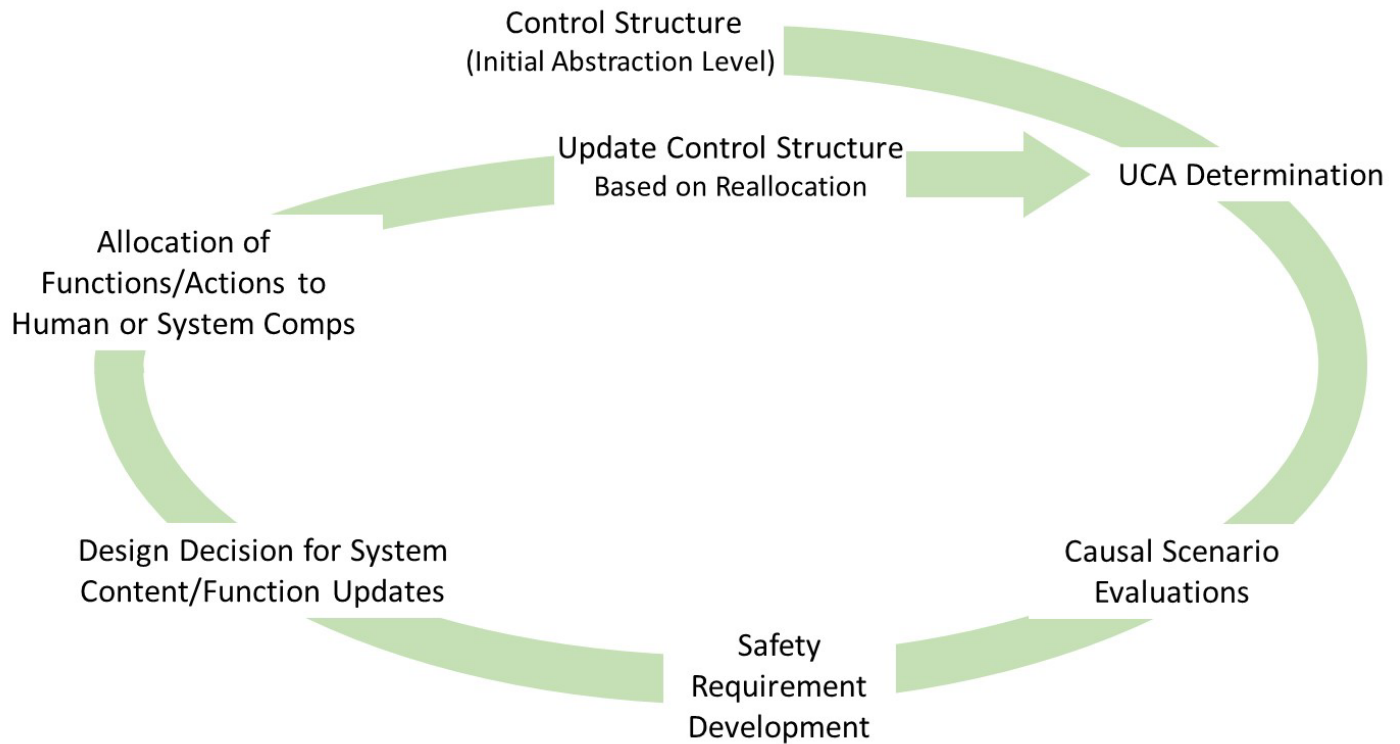


Figure 10 - Iterative cycle using STPA for system control structure updates

Therefore, a sequence with the following structure is constructed to maintain traceability:

UCA → Scenario → Safety Requirement → Design Decision →
Allocation of functions/control actions to human or system components →
Traceability to a new functional control structure.

Ideally, STPA analysis begins with the start of the system conceptualization. When STPA starts after system design has already started, a decision needs to be made to either:

- Incorporate existing design details into the analysis, using analysis results to impact design change retroactively at whatever stage of the design it is in; or
- Ignore existing design/implementation details initially, first implementing an abstract analysis independent of the detailed system design.

Once STPA analysis is done, then a gap analysis can be implemented for the existing design and any other architecture designed for the same purpose. The advantage of this approach allows for the STPA analysis to be reused for multiple architectures.

Expand the control diagram hierarchically, adding detail to each sub-level starting from the basic top-level used to help understand the system and related elements.

6.9 Define Unsafe/Unwanted/Unexpected Control Actions (UCAs)

An unsafe control action (UCA) is a control action that will lead to a hazard in a particular context and worst-case environment. The term “unsafe” refers to the hazards identified in STPA. As discussed earlier, hazards can include issues related to the loss of human life or injury (traditional safety). Still, losses can also be defined much more broadly to include other losses like a mission loss, loss of performance, environmental losses, and more (Leveson and Thomas, 2018). Events leading to non-safety related losses could be documented as a risk and considered a type of hazard.

STPA identifies unsafe control actions that may lead to hazards. The STPA Handbook (Leveson and Thomas, 2018) states that there are four ways a control action can be unsafe:

1. Not providing control action leads to a hazard.
2. Providing control action leads to a hazard.
3. Providing a potentially safe control action but too early, too late, or in the wrong order.
4. The control action lasts too long or is stopped too soon (for continuous control actions, not discrete ones).

These four ways are organized into a tabular heading and each category is analyzed:

Control Action	Not providing causes hazard	Providing causes hazard	Too early, too late, out of order	Stopped too soon, applied too long
----------------	-----------------------------	-------------------------	-----------------------------------	------------------------------------

Figure 11 - Guideword headings as shown in STPA Handbook (Leveson and Thomas, 2018)

Used with permission from Leveson and Thomas. All other rights reserved.

Sub-categories are possible. For example, “providing causes hazard” can include subcategories like “control action is intermittent,” “control action is too high,” and “control action is too low.” The STPA Handbook (Leveson and Thomas, 2018) contains additional guidance about these categories and subcategories.

Some control actions may directly represent a function of the system element, like braking. In that case, the control actions can be analyzed as identified “functions” of the system element of interest defined when developing the control structure content. In addition, STPA practitioners have made some enhancements by providing some helpful phrases under each guideword column, looking like [Figure 12](#).

	"NOT Providing" Cause Hazard	"Providing" Cause Hazard	Incorrect Timing Incorrect Order	Stopped Too Soon Applied Too Long
Function (Responsibility)	Missing Not Followed	Providing When Not Expected Provided More/Less than Required Provided Content Results in Control Conflict	Provided Too Early/Late When Required Provided Before/After When Required Provided Content in Wrong Order Provided Opposite of What Expected	Providing Unstable or Oscillating Content Providing Truncated Content Providing Stuck Content

Figure 12 - Guideword table with additional helpful phrases (Ressler and Vernacchia, 2019)

Used with permission from Ressler and Vernacchia. All other rights reserved.

Because the UCAs are used to define when behavior should or should not be required, the UCAs cannot assume that the requirements are known, or if they are known, must not assume that they are correct or complete. For example, a UCA may not simply state “Controller X does not provide brake command when required” because this assumes that correct and complete requirements already exist. The same is true for functions—the UCAs cannot assume that the required or intended functions were safe. A UCA is not limited to compliance with designed, intended, or required behaviors. A UCA can describe any behaviors that leads to hazards, including designed, intended, and required behaviors. A UCA may not simply state “Controller X does not provide brake command when intended” or “Controller X provides brake command when not expected” because that assumes the intended and expected behaviors are always safe. Instead, the UCA context must define the underlying conditions that make the action unsafe without relying on requirements, intentions, and expectations. An example of a stronger UCA is: “Controller X does not provide brake command when a forward collision is imminent.” This UCA does not just rely on assumptions about intent, expectation, or existing requirements, but instead describes the underlying condition (imminent forward collision) that makes the lack of control action unsafe.

UCA format is a key factor for conducting an effective STPA evaluation. UCAs should contain five different parts. The first part is the “source,” which is the system element that produces the control action. The second part is the “type” of UCA that corresponds to one of the guide word column headings. The third part is the “control action” itself. The fourth part is the “context,” in which the control action is produced. And the final part is the “link” to the identified hazard(s). An example of a function that will provide hill hold capability when the vehicle comes to a stop on an incline and where the hazard would be unintended vehicle motion is shown in [Figure 13](#).

A UCA contains five parts:

UCA-2: BSCU Autobrake provides Brake command during a normal takeoff [H-4.3]
<Source> <Type> <Control Action> <Context> <Link to Hazards>

Figure 13 - Example of appropriate UCA format

Used with permission from Leveson and Thomas. All other rights reserved.

The STPA Handbook will note that “UCAs are often written with each part in the same order shown above, but in some cases, it may be clearer or more natural to use a different ordering. The ordering is not critical. The key point is that UCAs contain these parts.”

UCA development may become expansive for complex systems as more and more abstraction levels are evaluated. An STPA practitioner may be faced with either creating numerous specific UCAs according to all the possible operating contexts or creating fewer higher-level UCAs used to accommodate several similar use cases. While either will support the downstream development safety requirements based on causal scenario evaluations, the second approach has been shown to be more effective in handling the prospect of having so many specific UCAs at the beginning of the evaluation that the problem seems too daunting to approach.

The analysis should link each newly identified UCA to either a hazard, a risk, or both. This will help prioritize the safety related UCAs for the analysis effort and still keep track of risks that will require system requirements. Ultimately, the goal is to generate good safety and system-level requirements that should not conflict with one another. In the event the analysis reveals a conflict between a high-level system requirement and safety requirement, an engineering management decision may be warranted on how to proceed.

For each element’s control action or function, examine each UCA category to generate UCAs

- Identify if there is a need to iterate between UCAs and hazards.
 - Each UCA must be associated with a hazard. If a UCA is identified, but no appropriate hazard exists, this may indicate that a new hazard is needed in the analysis. In this sense, hazard identification and derivation of UCAs is an iterative process.
- Deriving the context for UCAs can be done systematically by identifying key high-level inputs necessary for the control action or function. For instance, this might include the operator’s input to the system and inputs from interfacing systems. Each combination of inputs could then be assessed in combination with the control action/function and UCA category.
- Remove UCAs not associated with any hazard.
 - Highlight those UCAs for further STPA analysis to identify constraints for risks/threats and other losses that they could be linked to
 - Some UCAs could be related to overall system performance or the intended mission, so they should not be removed, just linked to other hazards/losses and highlighted for further analysis.
- Develop constraints for UCAs using “not” logic.

- Identify UCAs through a systematic reference of every control signal in the control diagram, using the four STPA criteria for these signals:
 - Safe action not provided.
 - Unsafe action provided.
 - Incorrect timing or order.
 - The action stopped too soon.
- Reference system strategy/focus to identify UCAs that have the most consequence.
- Standardize words and phrases used to define CAs and UCAs, with the expectation of duplication and looking for automation opportunities with available STPA tools. Consider using editing tools such as LaTeX to program standard phrases. Standardizing phrases not only helps with automation but also reduces error and adds clarity for finding unique CAs.
- For traceability, use unique labels for each UCA.
- Make sure the control diagram captures all CAs. For instance, if there is a CA for the driver to “depress the accelerator pedal,” the analyst should also consider CAs to “release the accelerator pedal.”
- Expect to refine the control diagram as UCAs are determined and additional signals and/or elements are identified.

As UCAs are determined and additional signals and/or elements are identified, expect to refine the control diagram.

6.10 Define Casual Scenarios

This step explores the possible causal scenarios that could lead to the identified unsafe control actions. Information is generated to assist designers in eliminating or mitigating the potential causal scenarios of the hazard. This involves examining the control loops and their parts and identifying how they could lead to undesired control action.

Causal scenarios are not faulted states such as those in FMEAs, although the most basic causal scenarios may be similar to faults identified through FMEA. In FMEAs, known failures are evaluated and their effects are assessed, so the key question is “what” failed. There may not be any failure in causal scenarios but rather a set of conditions or circumstances that could lead to a situation where a UCA might occur.

The key question in causal scenario evaluation is to ask “why.” This leads to two types of questions: first, “why” would UCAs occur; second, “why” would control actions be improperly executed or not executed, leading to a hazard. The STPA Handbook summarizes this point as follows and provides a graphic (Figure 2.17 from the Handbook shown in [Figure 14](#) below) illustrating the application of these scenarios to a control structure (Leveson and Thomas, 2018).

Two types of loss scenarios must be considered:

- a. Why would unsafe control actions occur?
- b. Why would control actions be improperly executed or not executed, leading to hazards?

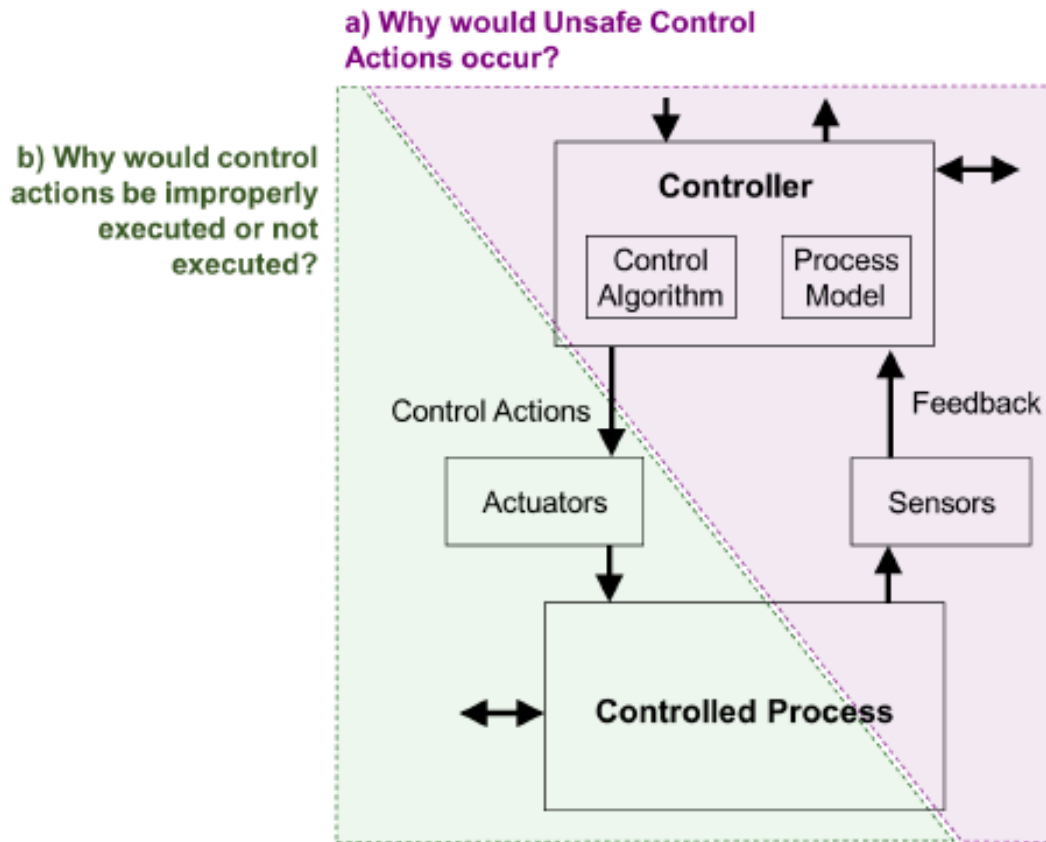


Figure 2.17: Two types of scenarios that must be considered

Figure 14 - Causal scenario graphic from STPA Handbook (Leveson and Thomas, 2018)
 Used with permission from Leveson and Thomas. All other rights reserved.

The two types of loss scenarios can be expanded by leveraging the previously mentioned “why” aspect of the causal scenario evaluation and breaking up control structure into four areas. These areas are listed here, with examples provided.

- Inappropriate decisions—Type (a) from above:
 - Controller micro-processor issues.
 - Poorly designed algorithms.
 - System changes that make process model obsolete.
- Inadequate feedback and other inputs—Type (a) from above:
 - The sampling frequency of sensors is inadequate.
 - Conflicting feedback inputs.
 - Incorrect feedback.

- Inadequate control execution—Type (b) from above:
 - Actuators receive conflicting commands from multiple controllers.
 - Actuators do not execute a command.
 - Actuator response time is inadequate.
- Inadequate process behavior—Type (b) from above:
 - The controlled process has other actuators from other systems affecting it.
 - The controller process does not respond to the actuator action.
 - The controller process does not execute the control action correctly.
 - Controlled process missing process element (e.g., high voltage power for electric powertrain).

Figure 15 shows a compressed natural gas (CNG) propulsion system control structure example with shaded areas depicting the two-major causal scenario type coverage. This CNG system can be powered by either natural gas or regular gasoline. The fuel injector control module (FICM) controls the flow of high-pressure CNG through a shutoff valve and then through an electronic pressure regulator that drops the CNG pressure to a level where the system fuel injectors can administer the fuel into the internal combustion engine. Fuel tank and fuel rail pressure sensors provide feedback to the FICM.

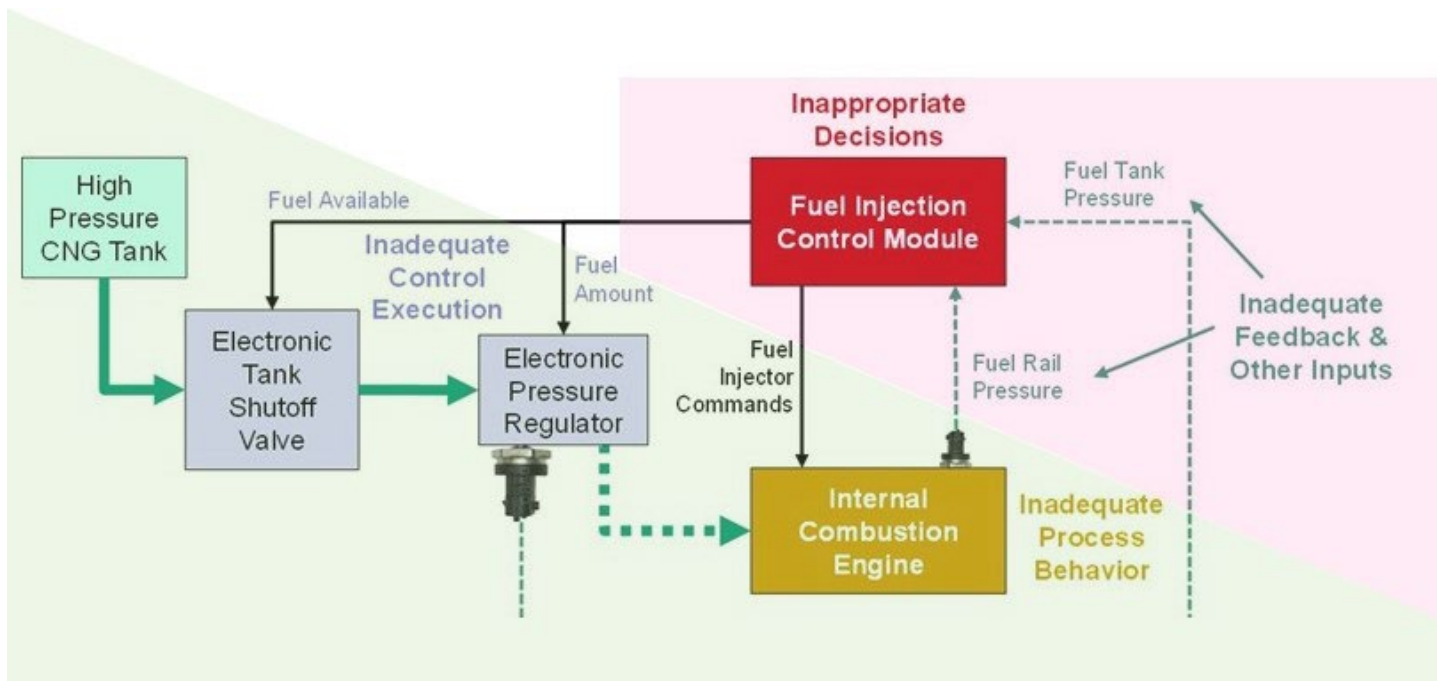


**Figure 15 - Compressed natural gas system:
causal scenario “type” areas (Ressler and Vernacchia, 2018)**
Used with permission from Ressler and Vernacchia. All other rights reserved.

Inspection of the four areas mentioned above could lead to identification of causes such as:

- Inappropriate decisions (fuel injection command module):
 - FICM did not realize the system was in gasoline mode.
 - FICM was designed to wait for driver input to switch fuel modes.
- Inadequate feedback and other inputs (sensors):
 - Low voltage to fuel tank pressure sensor.
 - Fuel tank pressure sensor resolution inadequate.
- Inadequate control execution (actuators):
 - fuel rails and injectors are over-pressurized.
 - Pressure regulator response time too slow.
 - Shutoff valve does not close completely.
- Inadequate process behavior (CNG engine):
 - Fuel injectors do not provide appropriate pressure.
 - Fuel injectors provide CNG when in gasoline mode.

These causes would appear in causal scenarios that explain how one or more causes come together leading to hazards and ultimately losses. Once causal scenarios have been evaluated, the STPA process moves on to identify the safety requirements necessary to prevent or manage these causal scenarios to an acceptable risk level.

Additional guidance for defining causal scenarios and causal factors includes:

- Define causal scenarios.
 - Identify scenarios that could lead to UCAs. (Why would UCAs occur?)
 - Bad “controller” behavior.
 - Inadequate feedback/information causes.
 - The key part of this is capturing system assumptions and ways they could be violated. If system assumptions are not detailed early on in the system scope and boundary, it will become apparent when trying to understand causal analysis and process model flaws, so capture all assumptions of system behavior early on.
 - Assess why commands/instructions (control actions) would have problems being conducted or not performed.
 - Control path issues.
 - Controller process entity issue.

- For implementation, there needs to be a systematic process using STPA provided criteria to test every signal in the control diagram if it is: missing, inadequate, incorrect, delayed.
 - S1 = AV breaches minimum safe distance of the forward mobile object when braking force command not provided in time.
 - S2 = AV exposes passengers to unhealthy g-forces when incorrect braking force command is provided.
- Safety goal, UCAs constraints, and requirements gathering.
 - Using a parameterized approach to developing context for UCAs can help identify parameter combinations that lead to conflicts that need to be resolved through system design (e.g., not providing an action may lead to hazard H1, but providing the action may lead to a different hazard).
- Develop top-level requirements—the system should not do “X” under these conditions.
- Incorporation of safety requirements into appropriate technical documents for system content design, supplier-provided entities, and validation test cases and planning.
- Define causal factors.
 - For each scenario, consider behaviors throughout the control diagram, such as: controller (group 1), actuator (group 2), controlled process (group 3), sensors (group 4).
 - For S2 example:
 - Controller (group 1): Incorrect braking force due to a compute HW fault, threshold value memory fault... etc.
 - Actuator (group 2): None.
 - Controlled process (group 3): None.
 - Sensors (group 4): None.
 - Take a systematic approach for each group.
 - Provide a “rationale table” for each scenario analyzed for causal factors. This is critical to capture the “why” of the analysis for later reference.
 - Use causal factors to refine control constraints—iterative STPA process.
 - Develop top-level requirements—the system should not do “X” under these conditions.
 - Incorporation of safety requirements into appropriate technical documents for system content design, supplier-provided entities, and validation test cases and planning.

6.11 Creation of Safety Requirements

Safety requirements ensure the system will be able to prevent or manage identified causal scenarios to an acceptable risk level. Each causal scenario is evaluated over its potential operating conditions, and appropriate requirements are developed. An example of this effort was developed using the causes from the CNG system in the last section, where the requirements are shown in italic text (Ressler and Vernacchia, 2018). (Used with permission from Ressler and Vernacchia. All other rights reserved.)

- Inappropriate decisions (FICM):
 - FICM did not realize the system was in gasoline mode.
 - FICM was designed to wait for driver input to switch fuel modes.
 - *ECM shall send a secured request to FICM indicating GAS mode active.*
 - *FICM shall not activate CNG mode if ECM signal indicates a gas mode.*
 - *FICM shall automatically switch fuel modes when the existing mode is unavailable.*
- Inadequate feedback and other inputs (sensors):
 - Low voltage to fuel tank pressure sensor.
 - Fuel tank pressure sensor resolution inadequate.
 - *Fuel tank pressure (FTP) sensor voltage supply shall be monitored by FICM.*
 - *FICM shall disable CNG operation during if FTP sensor not available.*
 - *FTP sensor resolution shall be 3 psi over the full operating range.*
- Inadequate control execution (actuators):
 - Fuel rails and injectors are over-pressurized.
 - Pressure regulator response time too slow.
 - Shutoff valve does not close completely.
 - *Fuel rail pressure shall be monitored by the FICM during CNG operation.*
 - *Pressure regulator response time shall be a maximum of 25 ms.*
 - *Fuel tank pressure shall be monitored by the FICM after shutoff valve is commanded <OFF>.*
- Inadequate process behavior (CNG engine):
 - The controlled process has other actuators from other systems affecting it.
 - The fuel injectors do not provide proper pressure.
 - The fuel rail has a connector leak that leads to loss of fuel containment.
 - *The FICM shall have prioritization logic to deal with multiple actuator actions.*
 - *Fuel rail pressure shall be monitored by the FICM during CNG operation.*
 - *Fuel rail pressure shall be monitored by the FICM during gasoline mode by use of fuel rail pressure sensors.*

Solutions for causal factors of loss scenarios can be treated as new requirements or constraints. New requirements or constraints derived by STPA are incorporated into the system design and are useful for updating the system architecture. The results of the STPA analysis should record traceability as a rationale for the derived requirements.

Management of requirements traceability between those requirements generated by an STPA process and those requirements generated from other processes is an important activity. For example, requirement information may be managed by updating MBSE system models or reflecting it in a requirements management tool.

6.12 Lessons Learned

This section lists lessons learned that may be useful to STPA practitioners:

- Leverage systems engineering principles.
 - Guidance (or examples) on how to create control structure by using systems engineering foundations.
- Begin with simple control structures and then go deeper.
 - Explain how to capture the system or function and how to create a control structure as a recommendation.
 - Provide examples of control structure, hierarchical layered structure versus non-layered structure. This may help STPA beginners to understand which control structure is more verifiable.
- Use STPA early in the engineering design process.
- Identify and engage a “chief system architect” for the system under review if available.
- Use collateral from existing safety process tasks to avoid repeating in STPA. (For example, some companies use PHA and HARA to identify potential accidents and hazards and to assess their associated risk levels. This info can be inserted into the STPA process to avoid parallel development of the same info.)
- Leverage STPA outputs in verification and validation test plans and test cases.
- STPA can complement, rather than replace, other methods—independent check, look at safety problem from multiple viewpoints.
 - For example, HAZOP may identify hazards based on function, while STPA may identify hazards based on unsafe control actions. These hazard identification techniques can be performed independently. Comparing the results can (1) make sure neither method overlooked a hazard, and (2) help develop more rigorous definitions of hazards (i.e., what does the hazard cover/not cover).
- Incorporation of safety requirements into appropriate technical documents for system content design, supplier-provided entities, and validation test cases and planning.
- Visualize the loss scenarios.
 - Even if the UCA occurrence scenarios are complex due to interactive interference, it is possible to explain systematically in an easily understandable way by summarizing them from the control flow viewpoint. It is effective as a safety argument.

The visualized loss scenario is helpful to plan the verification and validation, such as to prepare the testing environments and generate test cases

6.13 Questions to Prepare for STPA

This section lists questions that may be useful to STPA practitioners to prepare for an STPA evaluation.

- What is the goal or mission of the system?
- What does the system look like from an architecture, functional, process, or controls structure perspective?
- Is there an operational description of the system's expected functions, behaviors, capabilities, and interactions with other systems, users, and human operators over expected operating scenarios?
- What are the required and expected functions for each of the system elements?
- Is there a priority of command to be maintained between different sub-systems during normal operation? An example would be holding a vehicle stationary on a hill with brakes and then wanting to accelerate up the hill; when are brake commands prioritized above accelerator pedal commands?
- What accidents is the system capable of causing?
- What are the hazardous conditions that may lead to any of these accidents?

6.14 Questions While Performing STPA

This section lists questions that may be useful to STPA practitioners to perform an STPA evaluation.

- What happens to the system when a system element fails or misbehaves?
- Does the resulting system behavior create a potentially hazardous condition(s)?
- What are the risks and risk assessments associated with these potential hazards?
- Which part of the system assumes the failed/misbehaving element's functions?
- What are the resulting system capabilities after the failure/misbehavior?
- Do these resulting system capabilities create new/other potentially hazardous situations?
- Are these resulting system capabilities sufficient for achieving system goals/missions?
- What are the required diagnostic strategies and level of rigor necessary to detect failures or misbehaviors?
- What are the mitigation strategies and actions necessary to transition the system from a detected failed/misbehaving state to a known, acceptable safe state?
- Based on the severity and exposure factors of the risk assessment (and potential operator controllability where applicable), is the residual risk acceptable?

6.15 Lessons Learned for Incorporating STPA in Large Organizations

This section outlines some of the experiences certain STPA practitioners associated with the development of this document have had when trying to incorporate STPA into large organizations. The following items can be helpful when dealing with this effort (Vernacchia, 2019). (Used with permission from Vernacchia. All other rights reserved.)

- First and foremost, make sure there is a need STPA can fill (e.g., HMI - socio-technical benefits, etc.).
- Don't try to change the whole world . . . The goal should be not to solve world hunger, but just to feed the family.
- Maintain your vision . . . but be ready to modify based on good feedback or input.
- Leverage the idea of continuous improvement for existing processes by enhancing the use of systems engineering and systems thinking.
- Talk to other people inside and outside of your organization.
- Emphasize that STPA is a systematic approach to evaluate proposed system content for that content's ability to detect and mitigate a situation that could lead to potentially hazardous conditions.
- Propose STPA as an alternative to struggling methodology.
 - For example, use STPA as an alternative to DFMEA effort to deal with human factors.
- Operate below the "radar."
 - Be focused.
 - Do not alienate people with grandiose statements.
 - Be respectful of people's concerns.
- Look at other company successes (Boeing, Embraer, Ford, etc.).
- Seek out like-minded STPA practitioners in your industry or across industries to find common interests and needs.
 - SAE STPA Recommended Practices Task Force.
- Demonstrate the value of STPA requirements addressing safety concerns.
- Associate STPA with corporate initiatives when it helps those initiatives.
- Use on programs with new functions and features that have not been implemented yet or implemented together yet.
- Gather objective data showing results.
 - Requirements generated.
 - Design updates and changes are driven by STPA evaluations.
 - Short time to get results.

6.16 Japan Automotive Software Platform and Architecture (JASPAR) Lessons Learned

The STPA Task Force coordinated its efforts with other organizations using STPA in their functional safety processes. The following content is feedback from one such group, JASPAR, that has a Functional Safety workgroup whose purpose is to "improve functional safety development in terms of quality, workload, and speed in order to drive functional safety for enhancing vehicle system."

It is effective to define a general-purpose control structure to support the STPA introduction. Prepare generic control structures with different types of hierarchies and use them according to the purpose of analysis. For distributed development in complex and large-scale systems, the granularity of analysis is unified to promote common understanding and to communicate with stakeholders.

Examples of general-purpose STPA control structure in the automotive domain. The features are as described below.

1. Hierarchical control structure

Established a hierarchical control structure where vehicle level is ranked higher, and the system level is ranked lower, with two levels being interrelated.

2. Vehicle level control structures that incorporate comprehensive viewpoints

Components consist of driver/systems/environments/vehicles that should be considered a starting point for large-scale systems.

3. System-level control structures

The components are classified into different legends to clarify which elements relate to one's company and do not.

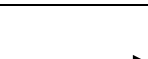
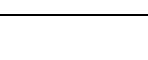
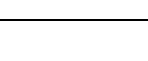
Example 1: If one's company's products are not relevant to the environment, descriptions of the environment are unnecessary.

Example 2: If there are interactions with ECUs made by other companies, use the components names of the other ECUs.

Example 3: Solid lines indicate interactions up to HMI, which relates to one's company's products, and dotted lines indicate the subsequent interactions between HMI and people.

6.16.1 Control Structure Legend

Table 2 - JASPAR lessons learned control structure legend

No	Shape	Explanation	Case
1		Human Controller	Driver, crew, dealer mechanic
2		System Controller	System, ECU, sensor, switch
3		Controlled process	Vehicle, mechanical parts, tires, brake pads, gearbox
4		Environment	Environmental factors (roads, pedestrians, weather, rear seat crews)
5		Interaction	Interaction
6		Related System	Handle as a black box (systems made by other companies, etc.)
7		Related Interaction	Assumed interactions (interactions involving systems made by other companies)
8		Contents of interaction	Driver operation ON/OFF

6.16.2 Vehicle Level Control Structure

The important thing is to start STPA from a layer with a high level of abstraction regardless of the system scale. In other words, that designating drivers, systems, vehicles, and environments as components adds a comprehensive field of vision to the analysis, and it is one of the general-purpose control structure features.

The relationship between the controller and controlled processes in a vehicle is expressed as a structure. Everything except for special systems such as airbags, etc., can be expressed using this general-purpose control structure. The relationship between the controller and the controlled processes is established between the driver and the systems, and a relationship is also established between the systems and the vehicle.

The environment is associated with the driver and system inputs as well as vehicle outputs.

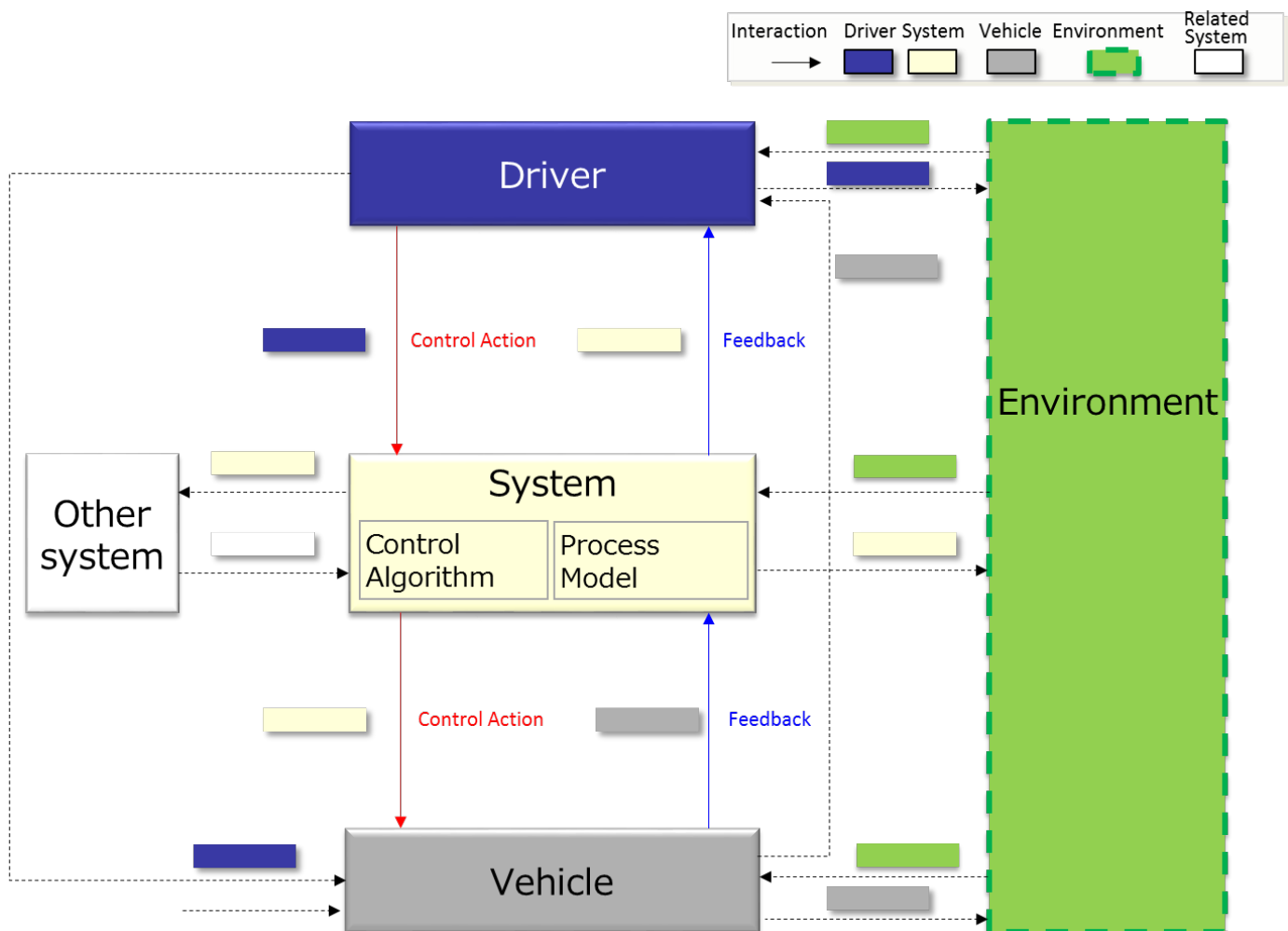


Figure 16 - JASPAR lessons learned vehicle level control structure

6.16.3 System-Level Control Structure

The relationship between the controller and controlled processes in the systems is also expressed as a structure. As it is necessary to describe the interactions even for components with no direct association with the relevant systems, these are expressed by changing the color of the system components. (See [Table 2](#).) The environment is associated with the driver and system inputs as well as vehicle outputs. If the relevant system has no association with the environment, the environmental components may be omitted.

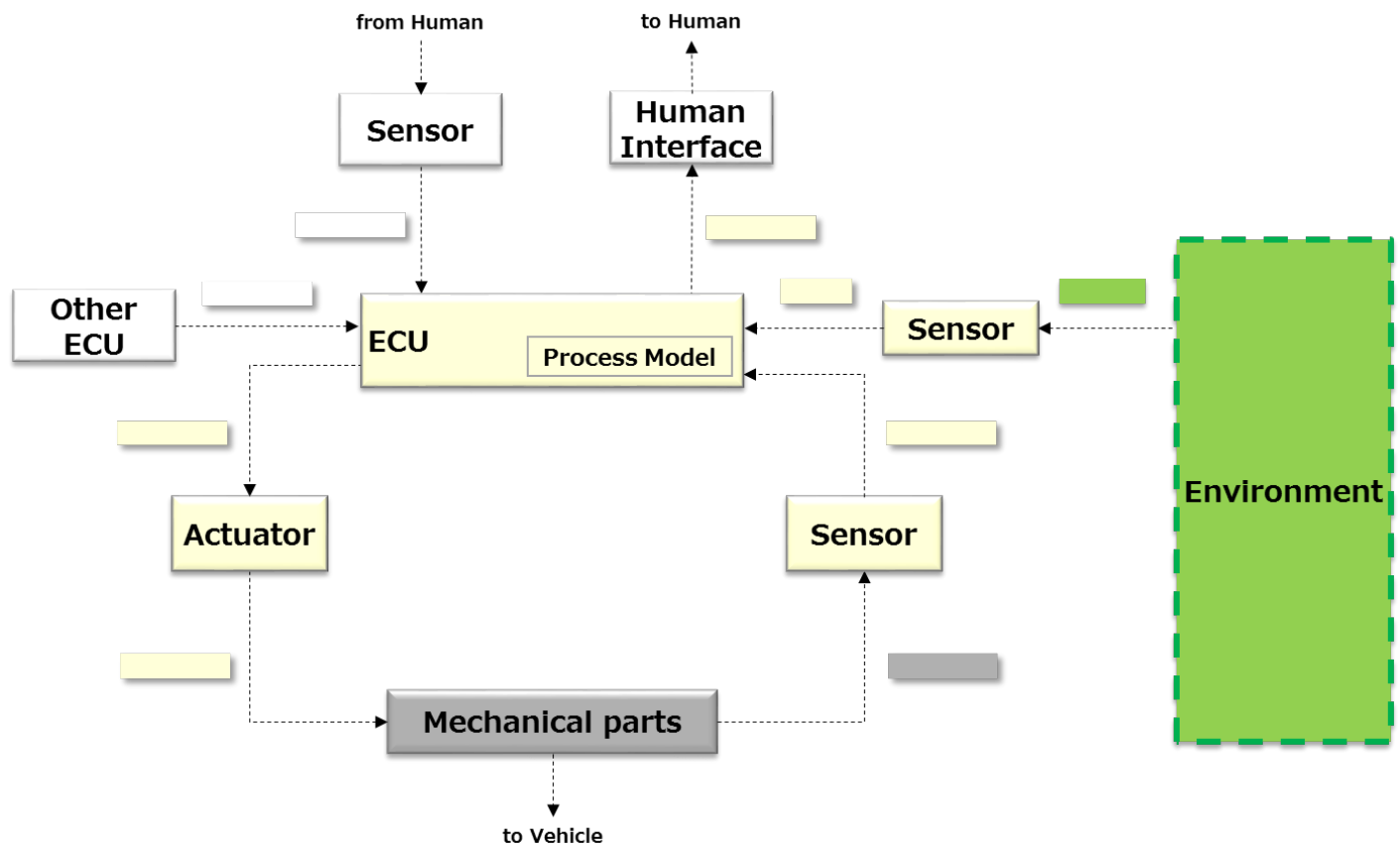


Figure 17 - JASPAR lessons learned system level control structure

When model-based systems engineering is adopted for system design, system models such as preliminary architecture can be referred to the control structure development.

Components of the control structure can be easily extracted from the system model, so a control structure can be constructed more efficiently.

The system's behavior (e.g., state machine diagrams, sequence diagrams, and activity diagrams of OMG SysML) helps to make communication smooth with reviewers and provide a foundation for the control structures and interactions of the analysis target systems.

7. SAFETY OF THE INTENDED FUNCTIONALITY (SOTIF) AND STPA

Considering a typical V-cycle of product development, one can consider the left-hand side of the V-cycle as the specification phase, while the right-hand side of the V-cycle is the testing phase. The specification phase involves identifying and illustrating the requirements at a system level, functional level, and unit level. The testing phase involves both the verification and validation of the requirements. In this section, we discuss how STPA can be used for both the specification phase and the testing phase. Refer to ISO/PAS 21448:2019.

7.1 Mapping STPA to SOTIF Clauses

In this section, we demonstrate the mapping of the STPA process to the various SOTIF clauses. Consistent with the V-cycle of product development, the SOTIF clauses can also be grouped into two phases: (1) specification phase, and (2) testing phase.

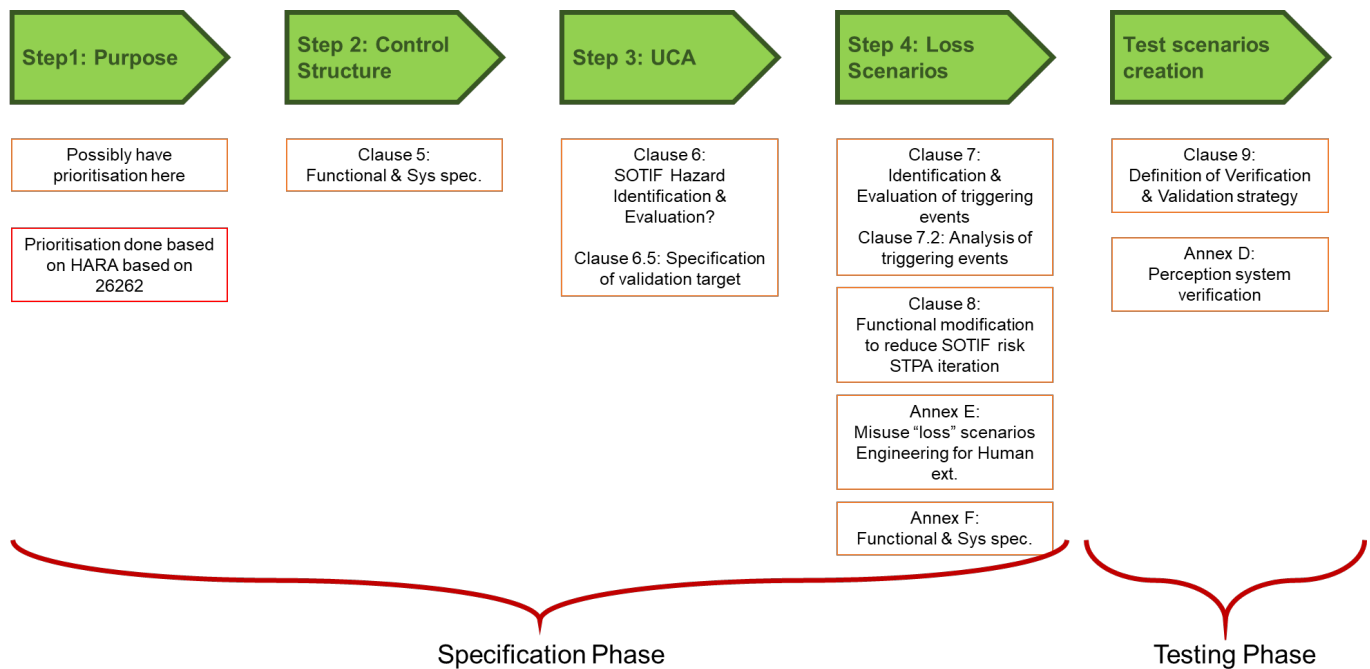


Figure 18 - Mapping Steps to SOTIF clauses

The specification phase involves identifying system and functional requirements, while the testing phase focuses on verification and validation involving the identification of test scenarios and safety targets. Steps 1 through 4 form part of the specification phase of the product development life cycle. Additionally, the testing phase of SOTIF specifies the need for identifying test scenarios and pass criteria. The outputs of Steps 1 through 4 can also be used to support this SOTIF need (i.e., the need for test scenarios and pass criteria) as STPA results could help identify those scenarios. [Figure 18](#) illustrates the mapping of the Steps to the SOTIF clauses. It is important to highlight that not all SOTIF clauses can be mapped to Steps 1 through 4 or test scenario generation based on Steps 1 through 4. Some aspects of SOTIF remain outside the scope of STPA, and additional methods will need to be adopted to address those clauses. Clauses not addressed by STPA include risk analysis, identification of validation targets ([6.5](#)), etc.

7.2 Unique Aspects of SOTIF for Specification and Testing Phase

It is suggested that to prove that automated driving systems (ADSs) are safer than human drivers, they will be required to be driven for more than 11 billion miles (Kalra and Paddock, 2016). However, even after driving 11 billion miles, one cannot make a claim for safety without considering the type of scenarios the ADS experienced in the driven miles. For example, if we drive an ADS for 11 billion miles on deserted roads, we have less to gain if our intended deployment area for the ADS is in an urban city center. Therefore, there is a widespread acknowledgment of the need to move away from a number of miles driven to the quality of miles, i.e., scenarios experienced in those miles.

One of the reasons behind suggesting the statistical argument for the driving requirement of 11 billion miles to prove the safety of ADSs is that it would potentially reveal all possible “black swan” events and make unknown scenarios/requirements or unexpected scenarios potentially leading to crashes, known ([Figure 19](#)). However, “black swan” scenarios by their very definition may or may not be discovered even with 11 billion miles of on-road testing. Therefore, rather than focusing on the number of miles, it is critically important to focus on methods to identify the black swan scenarios and corresponding requirements.

If we consider a quadrant approach to the scenario space, in addition to the known/unknown dimension, we may consider an additional dimension of safe/unsafe ([Figure 19](#)). From a SOTIF perspective, the SOTIF analysis aims to reduce unsafe quadrants (both known and unknown—Areas 2 and 3).

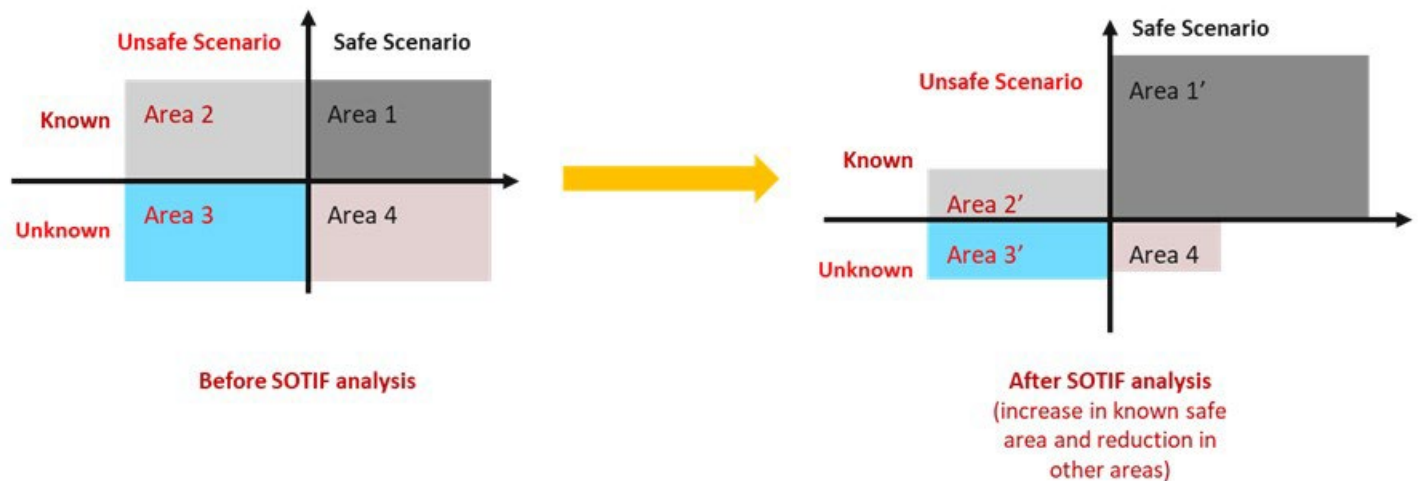


Figure 19 - Scenario quadrants with awareness and safety as axes

Extending the analogy from a safety perspective, ISO 21448 identifies four types of scenarios, as illustrated in [Figure 19](#). Area 1: known safe scenarios which represent known functionality that is also safe. Area 4: unknown safe scenarios which represent unknown functionality but still displaying safe behavior. Areas 2 and 3 are of interest from an STPA perspective. Area 2, which suggests known unsafe scenarios, represents known functionality but which displays unsafe behavior. Area 3, which suggests an unknown unsafe scenario, represents unknown functionality that displays unsafe behavior. The aim of the SOTIF process is to reduce the Area 2 and Area 3 quadrants ([Figure 19](#)) and move them to the Area 1 quadrant. STPA as a process enables the reduction of Area 3 by uncovering the unknown unsafe scenarios (and requirements). It also aids in evaluating the known functionality to establish if it displays safe or unsafe behavior.

Although requirements-based testing captures the “known knowns” efficiently, the inability to ensure its completeness leads to the occurrence of “unknown knowns,” “known unknowns,” and the “unknown unknowns” scenarios (e.g., black swan events). The latter three together could possibly be combined to identify hazardous scenarios that are not well understood. Defining such scenarios should be the goal for testers and identifying requirements for such scenarios should be the goal for specifiers. Additionally, [Figure 20](#) shows the distribution of scenarios around two different axes: understanding and awareness. Here the “known knowns” scenarios (or requirements) suggest scenarios that we understand as well as are aware of. While “unknown unknown” scenarios suggest scenarios that we are neither aware of nor are understood. The goal of a SOTIF methodology is to increase the “known knowns” quadrant while reducing the size of the other three quadrants. It is important to highlight that one cannot claim to have reduced the other three quadrants to zero. There will always remain a level of residual risk (both in terms of requirements as well as test scenarios) which will need to be argued and mitigated as part of the safety case for the ADS for a particular deployment.

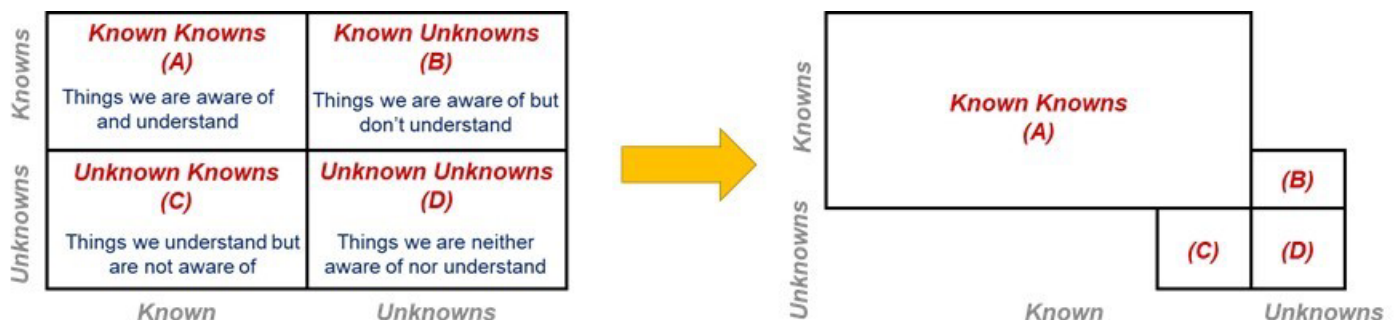


Figure 20 - Scenario quadrant with understanding and awareness as axes

Sections 6 and 8 illustrate the use of STPA as part of the specification phase for non-SOTIF applications. From a SOTIF perspective, STPA can also be used in the specification phase by using analysis to postulate potential loss scenarios for which SOTIF mitigation measures can be developed. That is, applying STPA provides a structured method that could potentially expose previously unknown scenarios (thus making them known). This application of STPA would be similar to the approaches described in Section 6 for sensor, algorithm, and actuator limitations and Section 8 for foreseeable misuse scenarios.

In addition, STPA can additionally support testing for SOTIF by providing detailed loss scenarios that can be incorporated into road testing and simulation. In particular, STPA can provide the traceability between these testing scenarios, unwanted system behavior (i.e., UCAs), and resulting vehicle-level hazards

7.3 Specification Phase: Using STPA for Requirements Identification

STPA Step 1 involves defining the purpose of the analysis. This involves the definition of losses and hazards, which form the basis of the analysis. It is possible to rank the losses and hazards to enable a ranking in the subsequent STPA artifacts (unsafe control actions and loss scenarios). STPA Step 2, which involves articulating the control structure for the system, helps identify the sub-system interactions and capture missing or superfluous functional modules/blocks. Insights from STPA Step 2 can be used as part of the functional and system specification (SOTIF ISO 21448 Clause 5). SOTIF Clause 6 involves SOTIF hazard identification and evaluation. STPA Step 3 can cover part of Clause 6 where UCAs are identified. STPA doesn't cover the risk evaluation part of SOTIF Clause 6. STPA Step 4 involves identifying "loss scenarios," which are essentially the triggering events for the UCAs (i.e., hazardous events). Thus, STPA Step 4 helps meet Clause 7 of SOTIF ISO 21448 (identifying and evaluating triggering events). It is important to highlight that the context of the UCA is a key component for the analysis, which makes a control action safe or unsafe.

Applying STPA to a system is an iterative process. Based on the results from the first iteration of the analysis, requirements will be identified leading to the addition of control actions or functional modules. It must be noted that deriving requirements is a by-product of the STPA process. This will then require a second iteration of the analysis. Thus, the functional modification to meet the new requirements (which in turn is to reduce the risk) is in line with SOTIF ISO 21448 Clause 8.

7.4 Identification of Relevant Control Actions

In order to identify relevant control actions (i.e., control actions of interest) in STPA Step 2, one may take an ODD-based approach. The ODD represents the operating environment within which an ADS can safely perform the dynamic driving task (DDT). SAE J3016 defines ODD as "Operating conditions under which a given driving automation system or feature thereof is specifically designed to function, including, but not limited to, environmental, geographical, and time-of-day restrictions, and/or the requisite presence or absence of certain traffic or roadway characteristics."

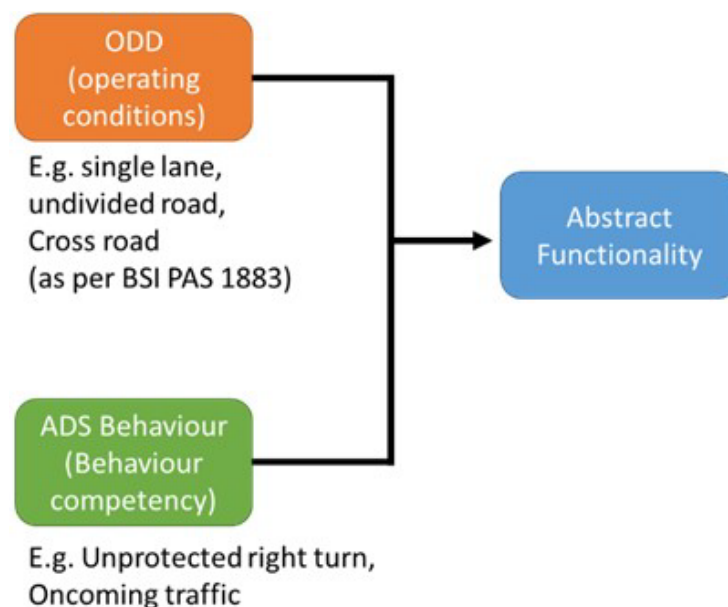


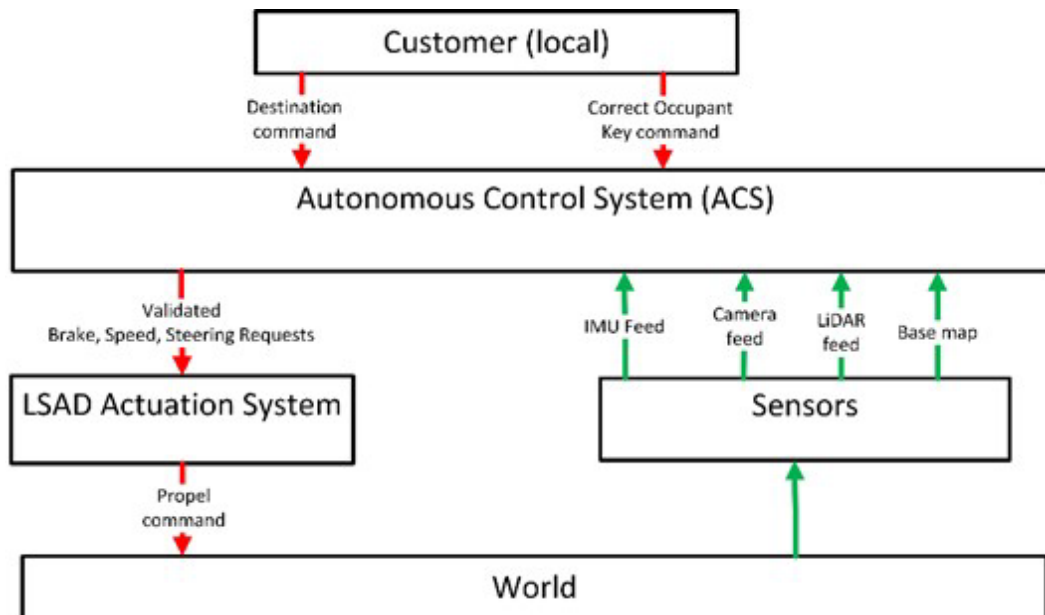
Figure 21 - ODD and ADS behavior with abstract functionality



Table 3 - Functionality with ODD attributes and behavior competency (example specification)

Understanding Functionality	ODD Attributes	Behavior Competency
	Motorway	Lane center
	Four lane	Lane change
	No hard shoulder	Overtake
	Variable road signs	
Potential Functional Modules	Variable sign detector, lane change module, lane marking detector	

Additionally, the ODD-based approach could be combined with a behavior competency concept to identify additional control actions of interest. Once an ODD and expected behavior has been identified (i.e., functionality of the system), a high-level control architecture can be developed (e.g., [Figure 22](#)).

**Figure 22 - High-level control architecture of an automated driving system**

7.5 SOTIF Clause 6: SOTIF Hazard Identification and Risk Evaluation

One important consideration for the hazard identification process is integrating some form of risk assessment to understand whether specific hazards or traceable UCAs or scenarios may have unreasonable severity or probability of occurrence. Traditionally, risk assessment techniques are utilized to categorize risk levels of identified hazards during the preliminary phases of an engineering program. One example of this approach is preliminary hazard analysis (PHA). PHA is a guided effort for identifying hazards, their associated causal factors, effects, and level of risk (Vincoli, 2005). A derivative of PHA for the automotive world is the ASIL levels defined in ISO 26262. These levels can be used to classify risks associated with a hazard. One of the primary questions for STPA and its impact on risk assessments is the degree of overlap between these approaches.

Although these approaches typically tend to address different aspects of system safety engineering, STPA can be used to inform the risk assessment. One example of this approach is shown by Pope (2019). Their objective was to identify whether STPA could identify risks within a managerial control structure and subsequently assess the risks. The case study applied STPA and identified 84 risks that were then classified. Each risk had an associated mitigation and risk response, indicating whether the risk needed to be mitigated immediately or simply monitored over time. This example identifies how STPA can be used in a complementary fashion with risk assessment procedures to ensure all hazards are assessed appropriately. As mentioned earlier, the STPA process cannot (like any other hazard identification method) guarantee a complete removal of Area 2 and Area 3. This will inevitably leave behind a residual risk.

7.6 SOTIF Clause 9: Definition of Verification and Validation Strategy

Completeness of any hazard analysis is a difficult task. Showing that a safety analysis comprehensively and completely covers all possible hazards may not be possible, but several properties of STPA help address the concern. One of these is the method's top-down refinement of losses and hazards. Completeness can be ensured if the initial, high-level list of hazards is complete, and each refinement step breaks them into collectively exhaustive subsets. Verifying completeness of this top-down process is ultimately the responsibility of subject matter experts, and STPA's hierarchical approach assists by providing traceability of low-level, detailed causal mechanisms back to their hazards.

STPA also helps to ensure completeness via a systematic approach for identifying unsafe control actions. Systematic generation of various unsafe control actions reduces the possibility of overlooking accident mechanisms and underestimating risk. An example of working through this process is highlighted in an STPA study of the FAA's NextGen traffic management system (Fleming et al., 2013). This report provides an example of automatically generating unsafe control actions for Air Traffic Control towers and performing a comparison with subject matter expert understanding of the control actions. The study noted that using software tools to generate unsafe control actions can ensure a comprehensive analysis; however, this does require experts to analyze the results and validate which unsafe control actions are applicable depending on the hazards of interest.

7.7 Testing Phase

For ADASs and ADSs, it is suggested that it is more important to identify "how a systems fails (or misbehaves)" as compared to "how a system works" (Khastgir et al., 2018). The purpose of SOTIF is not about failures but rather misbehavior. This is also true from a SOTIF perspective whose focus is on Area 2 and Area 3. Subsequently, in order to identify the "interesting scenarios" (i.e., black swans), a hazard based testing (HBT) approach may be used to complement the traditional requirements-based testing approach used in the automotive domain. With a focus on failures (and misbehavior), HBT has the potential to identify the "interesting" scenarios. Inspired by the aims of the SOTIF testing phase, the aim of using hazard-based test scenarios is to increase the "known knowns" space and decrease the "unknown unknowns" space. An HBT approach has the following steps (Khastgir et al., 2021):

- Identification of hazardous events (akin to STPA loss scenarios identified in STPA Step 4).
- Creating test scenarios for the identified hazard (high-level hazards are identified in STPA Step 1. However, test scenarios are identified after completing STPA Step 4 as it uses the outputs of STPA Steps 1 through 4).
- Pass criteria for the created test scenarios.

It is important to note that STPA may also be used in systems with human-machine interaction to uncover unsafe or misuse scenarios. STPA in the context of HMI is discussed in Section [8](#) of this document.

7.8 Testing Phase: Using STPA for Scenario Generation

This section discusses a methodology to create test scenarios from the STPA output (Steps 1 through 4) which have been illustrated in Khastgir et al. (2021). This involves identifying test scenario parameters for various elements of STPA. A complete test scenario definition should also contain the "pass criteria" for the corresponding test scenario from a practical usage perspective. With this understanding, a complete test scenario description will have five components:

- Scenery.
- Environment.
- Dynamic elements.
- Additional context.
- Pass criteria.

7.8.1 Scenery, Environment, and Dynamic Elements

Scenery defines all geo-spatially stationary objects in the operational design domain (ODD) of the vehicle. All moving objects and actors in the world comprise the dynamic elements category. Various attributes of scenery and dynamic elements can be identified from the BSI PAS 1883 taxonomy for ODD or SAE AVSC Lexicon or any other ODD taxonomy standard. The scenery can be classified into drivable areas, junctions, zones, fixed road structures, temporary road structures, and special structure attributes at the top-level. While this section (and the methodology discussed here) focuses on an automotive application, the methodology can also be extended to an aerospace or marine application.

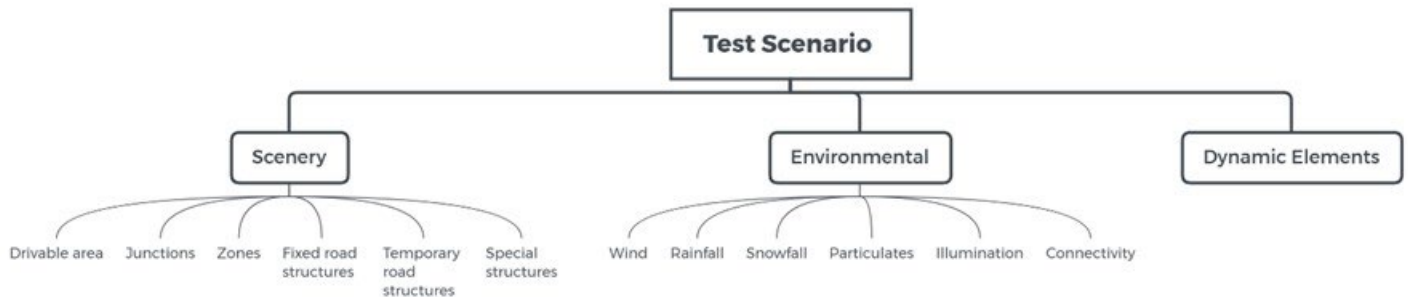


Figure 23 - Scenery, environmental, and dynamic elements of a test scenario

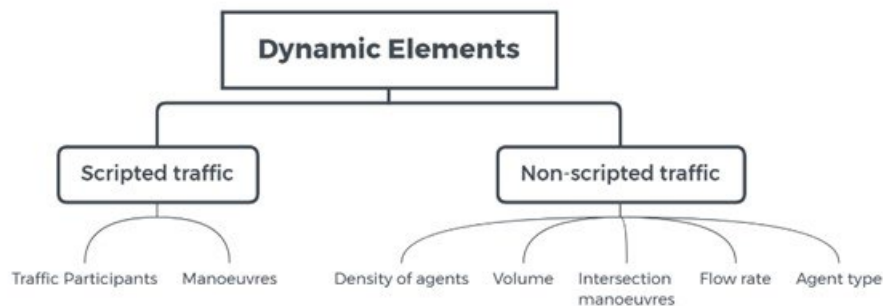


Figure 24 - Dynamic elements components

The scenery and the dynamic elements for the test scenario can be selected based on the defined Operational Design Domain (ODD) of the ADS (in STPA Step 1) (base parameters). Additionally, while we are discussing the testing phase in this section, ODD attributes may play a key role in the “context” definition of the UCAs in STPA Step 3 and the loss scenarios. For example, a control action (like braking) may be unsafe when provided on a high-speed motorway but may not be unsafe on a busy business park road. In another example, a UCA might suggest “provides braking too late when on a city road and pedestrians are present.” In contrast, the corresponding loss scenario might include dynamic elements like pedestrians entering the road outside of a crosswalk and scenery features like other vehicles obscure pedestrians from the ADS. Thus, in the automotive context, we suggest an additional step in STPA Step 1, which involves the definition of the operational design domain of the system. The ODD definition may be defined as per industry standards like BSI PAS 1883, ISO 34503, SAE AVSC Lexicon, etc.

The pass criterion defines the set of conditions (internal to SUT or external to SUT) for which the test scenario will be considered a pass. Additional context refers to the context element of the UCA (STPA Step 3) and the causal factors for “potential control action not followed/how could this happen” in STPA Step 4.

7.8.2 Additional Context

Additional context refers to additional parameters (of interest) for the test scenarios, which are derived from Khastgir et al. (2021):

- Context elements of the UCA (these may or may not be defined as part of the ODD and OEDR).
- Causal factors from STPA Step 4.

Some examples of additional context include ego vehicle or system states, external control inputs (e.g., remote operator), sensor delay time, etc. While performing virtual or real-world testing, parameters derived from “additional context” may potentially be used at a much higher resolution (higher priority) than other ODD or OEDR parameters. This is the reason for treating additional context as a special set of test parameters.

As discussed in Section 6, the unsafe control action (UCA) is structured into five components. As an example, let us consider the UCA—“GPP doesn’t provide waypoint design area when there are obstacles on the path and pod is moving (link to hazard).” Here, GPP refers to “global path planner,” and “pod” refers to a low-speed automated driving system equipped vehicle, i.e., shuttle.

As mentioned earlier, the additional context comes from both STPA Step 3 (from the UCA) and STPA Step 4. One of the parameters identified for the additional context is the “context” of a UCA. In the earlier UCA example, the content element is “...when there are obstacles on the path and pod is moving.” The corresponding parametrization of “pod is moving” and “overlapping distance of obstacle and the pod path” is done with respect to pod trajectory and pod velocity. Overlapping distance, trajectory, and velocity form the additional context parameters for the test scenario.

7.8.3 Pass Criteria/Validation Targets

Consider a control loop (with the controller with process model and control algorithm, controlled process, actuator, and a sensor) (Figure 25). For a UCA to occur, the process model of the controller could represent a belief that makes the control action it is directing appear to be safe (when it is unsafe, i.e., a UCA). Let us call this belief B1. If B1 were not true, the controller would not direct the original control action (i.e., original UCA). Therefore, one of the pass criteria for the test scenario would be defined as the negation of the belief B1 (process model belief), i.e., B1', as identified in STPA Step 4. Secondly, the process model has the belief (B1) because of some reasons. Let us call these reasons causing the process model belief B1 as B2. Once again, if these reasons B2 were not true, B1 would not be true (when treated recursively), and subsequently, the controller would not direct the UCA. Thus, the second pass criterion for the test scenario is the negation of the reasons for the process model belief, i.e., B2' (Khastgir et al., 2021). Thus, the two-pass criteria for the test scenario coming out of STPA Step 4 for each UCA are B1' and B2'. Therefore, the pass criteria for the test scenarios is the negation of the “process model belief” and negation of the “reason for the process model belief.” This includes the analysis of the fact that the UCA is not an issue and the fact that it is not issued for the right reasons (i.e., preventing an incorrect process model).

One could also view the identification of the pass criteria as requirements for functional modifications or SOTIF functional modification goals. For example, not having B1 might equate to improvements to the perception and decision algorithms; and not having B2 might improve sensors' improvements. For a particular scenario like delays in sensor feed (see Table 5), one could consider passing criteria 1 as “OD classifier shall not believe that the obstacle is not in vehicle trajectory (i.e., pose and surface probability don't collide)” and pass criteria 2 as “covariance error shall not be low.”

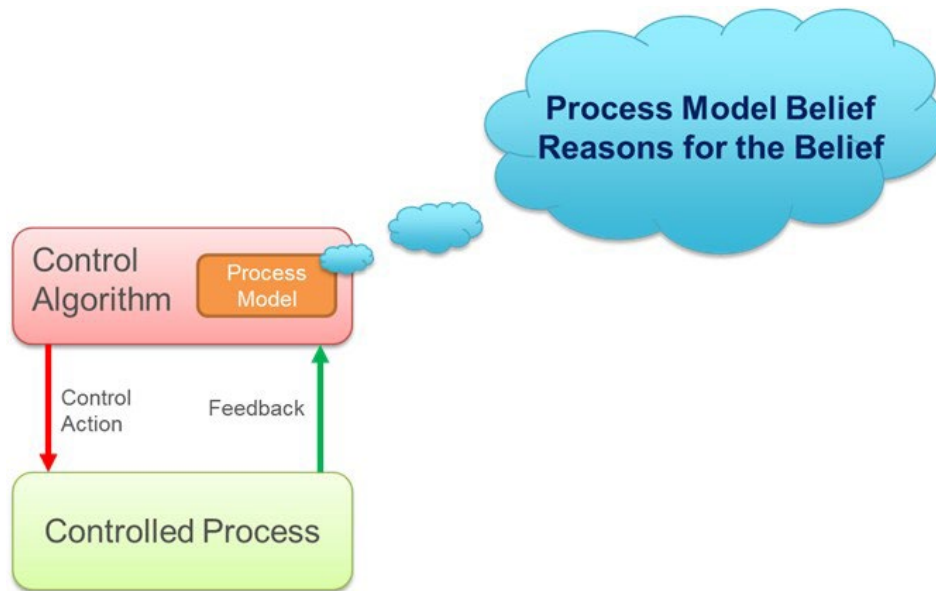


Figure 25 - Control loop with process model beliefs

Additionally, the “process model belief” and the “reasons for the process model belief” also provide us with leading indicators to monitor the system and serve as a precursor for an unfolding hazardous situation.

One may also consider that a possible mitigation strategy to prevent the UCAs from occurring may include monitoring the assumptions. This has been suggested as “assumptions based leading indicators” (Leveson, 2015). By incorporating assumptions-based leading indicators, we can identify additional system requirements as part of the mitigation strategy for the UCAs.

7.9 Case Study: Low-Speed Automated Driving System

This section discusses the specification and testing approach for a low-speed automated driving (LSAD) System (SAE Level 4) case study. The ODD for the LSAD system is an urban area on predefined routes. STPA Step 1 involves the identification of the losses and hazards. For the LSAD system, the identified losses and hazards include:



Figure 26 - Low-speed automated driving system
Image credit: Aurigo Driverless Technology

7.9.1 STPA Step 1

For the low-speed automated shuttle case study, the following losses and hazards have been identified:

- Losses:
 - Collision with objects outside the vehicle or damage to vehicle (L1).
 - Not completing the journey with passenger and cargo (L2).
 - Time of journey being too long, i.e., service target not met (L3).
 - Loss of life or serious injury to people or goods (L4).
- Hazards:
 - Vehicle does not maintain safe distance from nearby objects (H1)—Linked to L1.
 - Vehicle enters dangerous area/region (H2)—Linked to L1.
 - Vehicle exceeds safe operating envelope for environment (speed, lateral/longitudinal forces) (H3)—Linked to L1, L2, L3.
 - Vehicle occupants exposed to harmful effects and/or health hazards (e.g., fire, excessive temperature, inability to escape, door closes on passengers, etc.) (H4)—Linked to L4.
 - Vehicle does not follow an efficient, complete path to destination (H5)—Linked to L2, L3.

It is important to note that in the STPA process, the losses and the hazards need not necessarily be only associated with safety. However, the SOTIF process focuses on safety-related hazards. One can have losses and hazards associated with customer satisfaction, quality of service, etc.

7.9.2 STPA Step 2: Control Structure

As mentioned in Section 6, STPA Step 2 involves creating a control structure of the system under test (SUT). The most important aspect of the control structure development is the identification of the interactions (control actions and feedback) between the subsystems.

It is advisable to start at a higher level of abstraction, e.g., for an LSAD system, it might be done using five different components ([Figure 22](#)). The high-level control structure is made of five component. The first component is the world in which the LSAD is being deployed. The second component is sensor suite. The third component is the autonomous control system of the LSAD. The fourth component is the actuation system, which propels the vehicle. The fifth component is the human input. As the analysis proceeds, one can add additional details to the control structure by decomposing the individual blocks and using a lower abstraction level.

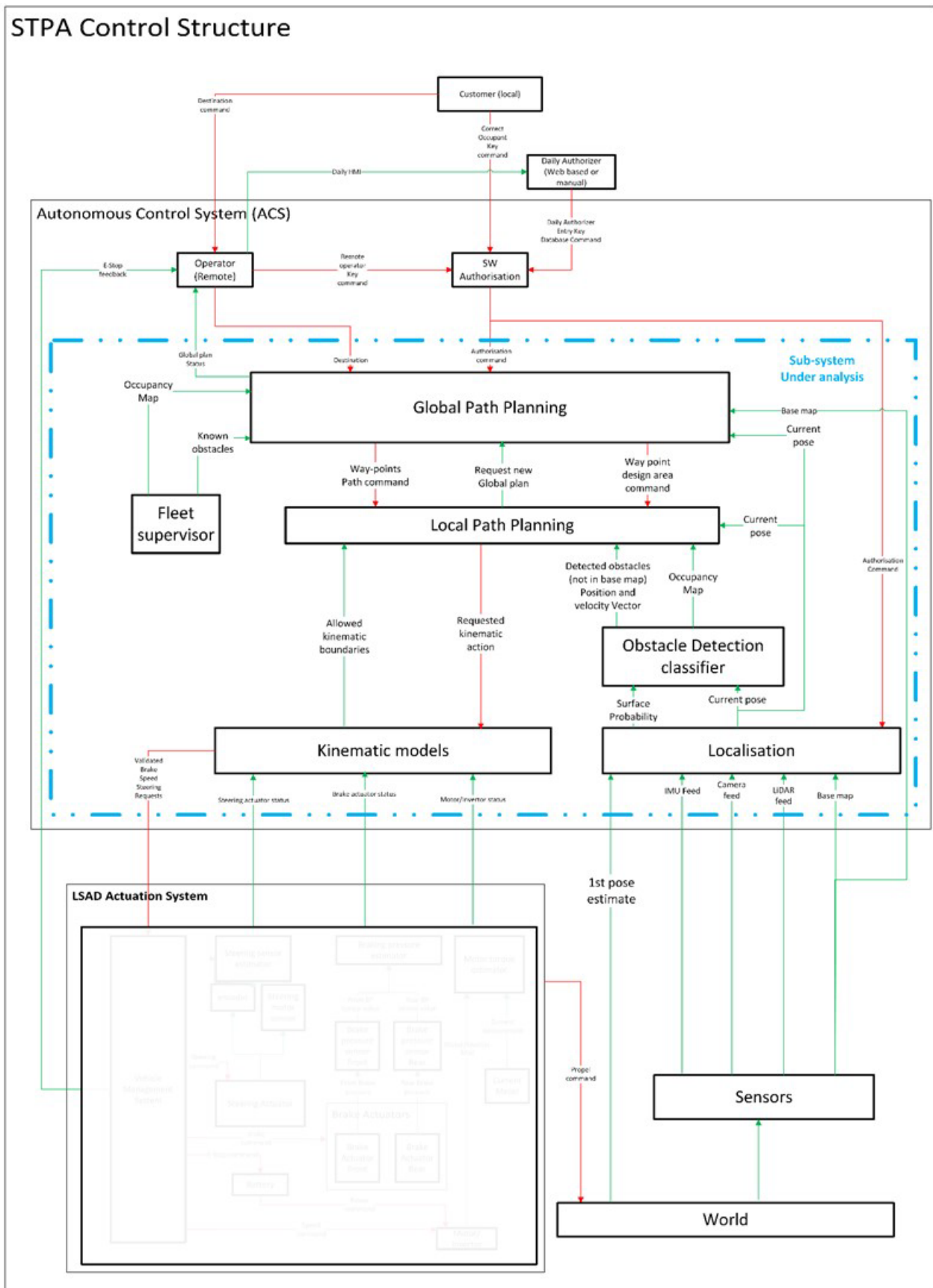


Figure 27 - Control structure for low-speed automated driving system case study

In this section, the control actions and feedback identified in [Figure 27](#) are discussed in detail.

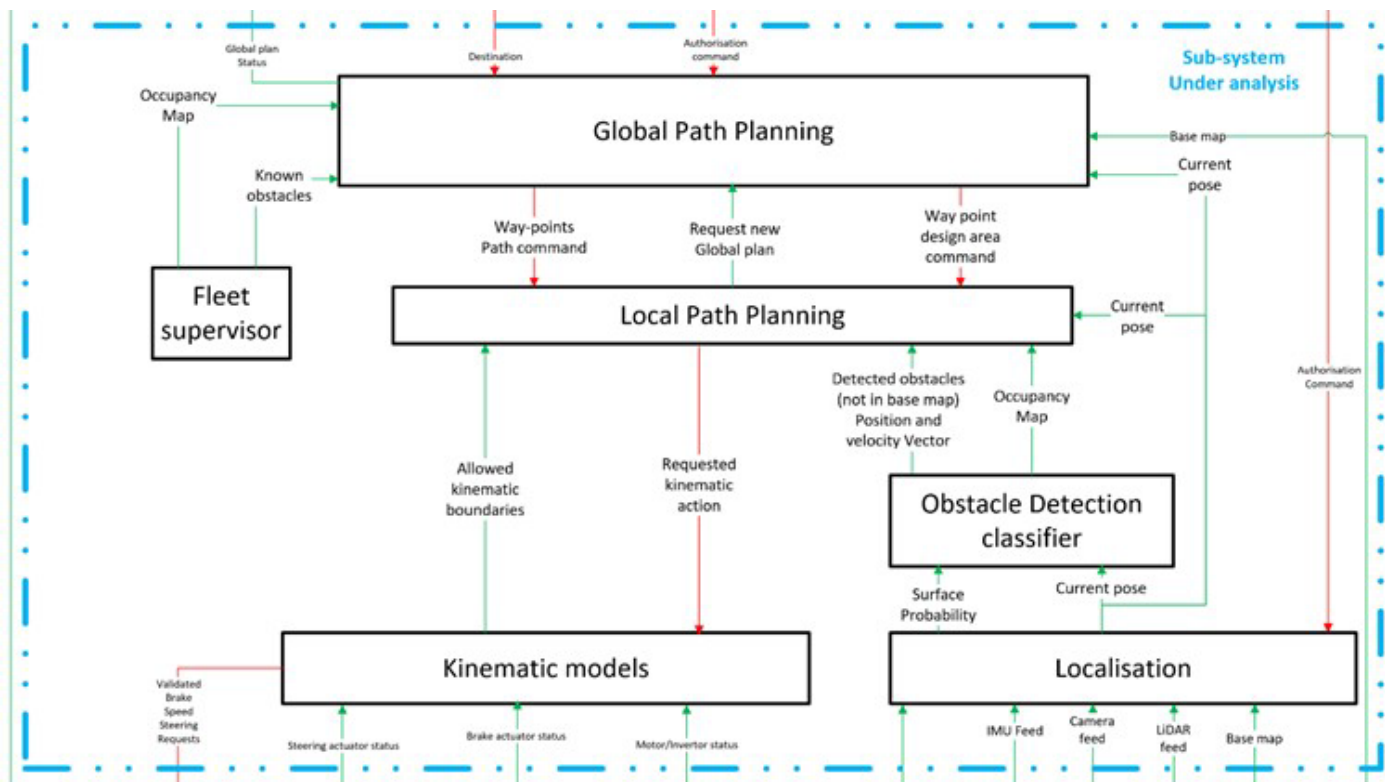


Figure 28 - Sub-system under analysis for the case study

7.9.3 STPA Step 3: Unsafe Control Action

Table 4 - Set of UCAs for the case study

Control Action	Not Providing Causes a Loss	Providing Causes a Loss	Too Early, Too Late, Out of Sequence Causes a Loss	Stopped Too Soon or Applied Too Long Causes a Loss
Destination command	[UCA# 4a] Occupant doesn't provide destination command when occupant needs to undertake a journey. [H5] [..]	[UCA# 4b] Occupant provides destination command when that is unachievable or presents a road hazard. [H2, H3, H5] [..]	[..]	[..]
Requested kinematic command	[UCA# 15a] Local Path Planning (LPP) doesn't provide kinematic action (braking) when there is a valid local path and the pod is moving and there is an obstacle in front. [H1, H2, H4, H5] [..]	[UCA# 15b1] LPP provides kinematic action (acceleration) when there is no valid local plan. [H1, H2,, H4, H5] [UCA# 15b2] LPP provides kinematic action when vehicle is not following the correct local plan. [H1, H2, H3, H5] [..]	[UCA# 15c1] LPP provides kinematic action (braking) too late after conflict is unavoidable when there is an obstacle in front and pod is moving. [H1, H2, H3, H4, H5] [..]	[UCA# 15d1] LPP stops kinematic action (steering) too soon before turn is complete when turning around a bend. [H1, H2, H4, H5] [..]

Depending on the ODD of the systems, the path along with the features on the path may vary. Similarly, the type of obstacles the system may encounter may also vary depending on the ODD of the system. For example, if a system's operating condition is a busy city share pathway, the features and entities will differ from that of a close business park.

7.9.4 STPA Step 4: Loss Scenarios

While there are many loss scenarios associated with each UCA, we focus on one example loss scenario in this section and illustrate the corresponding test scenario generation process.

- UCA#15a: Local path planning (LPP) doesn't provide kinematic action (braking) when there is a valid local path and the pod is moving and there is an obstacle in front. [H1, H2, H4, H5]
- Process model believes (B1): Three different process model beliefs to trigger the UCA to happen:
 - LPP believes that obstacles are not in vehicle trajectory. (15aB1.1)
 - LPP believes that there is a safe distance between current position of the vehicle and the obstacle (which is on vehicle trajectory). (15aB1.2)
 - LPP believes that it is providing the kinematic action (braking). (15aB1.3)
 - Etc.