SafetyLens: Visual Data Analysis of Functional Safety of Vehicles

Arpit Narechania*
Georgia Institute of Technology

Ahsan Qamar†
Ford Motor Company

Alex Endert‡
Georgia Institute of Technology

Fig. 1. The SafetyLens Compare view: (A) Project Panel comprising the visualization canvas and three tabs: (1) Nodes, (2) Links, and (3) Trace, (B) Shared Nodes Panel with three nodes (Hazards) selected and highlighted across all three projects (ellipses), (C) Detail View Panel showing details of a node (System Behavior) hovered in Project-B («), and (D) opens into a Control Panel comprising search and filter controls.

Abstract—Modern automobiles have evolved from just being mechanical machines to having full-fledged electronics systems that enhance vehicle dynamics and driver experience. However, these complex hardware and software systems, if not properly designed, can experience failures that can compromise the safety of the vehicle, its occupants, and the surrounding environment. For example, a system to activate the brakes to avoid a collision saves lives when it functions properly, but could lead to tragic outcomes if the brakes were applied in a way that’s inconsistent with the design. Broadly speaking, the analysis performed to minimize such risks falls into a systems engineering domain called Functional Safety. In this paper, we present SafetyLens, a visual data analysis tool to assist engineers and analysts in analyzing automotive Functional Safety datasets. SafetyLens combines techniques including network exploration and visual comparison to help analysts perform domain-specific tasks. This paper presents the design study with domain experts that resulted in the design guidelines, the tool, and user feedback.

Index Terms—Visual data analysis, Design study, Network visualization, Functional safety, Automotive engineering.

1 INTRODUCTION

* e-mail: arpitnarechania@gatech.edu
† e-mail: aqamar2@ford.com
‡ e-mail: endert@cc.gatech.edu

Modern automobiles come integrated with a complex suite of systems (e.g., Anti-lock Braking System) that deploy hardware components integrated with millions of lines of software code. In a system that is not designed properly, the sheer number and complexity of components can result in failures posing risks which if not appropriately handled can be detrimental to the safety of the vehicle, its occupants, and the surrounding environment. Recalls can tarnish a manufacturers brand, straining customer trust and loyalty. It is important to design and manufacture vehicles in a way that may assist in reducing the risk. To analyze a given design, several systems engineering analysis techniques exist, which include but are not limited to Failure Mode Effect Analysis [52], Fault Tree Analysis [35], etc.

Our work here is focused on Functional Safety in the automotive
The above example showcases one scenario of the “Adaptive Cruise Control” system. There are other vehicle systems (e.g., Anti-lock Braking System, Parking Assist System) with their potential scenarios (similar or different sets of expected functions, associated risks, and safety goals). Finally, car companies have multiple car models, with updates from year to year, each with potentially different systems. This makes Functional Safety datasets large. Figure 2 illustrates the “Adaptive Cruise Control” scenario in the form of a network with each node representing an entity. The purple boxes indicate the corresponding Functional Safety terminologies for the entities in the example (described in Section 2.1).

Functional Safety domain experts are engineers and analysts who can identify and assess hazards, set goals to minimize risks associated with these hazards, and define safety requirements to achieve these goals. They must also ensure smooth implementation of these goals by asking questions such as “Are there any scenarios where there is an unreasonable risk for which there is no safety goal?”. “Are there hazards in the current project that were also addressed in another project? If so, what safety goals were identified for them?”. and others.

We designed and developed SafetyLens, a visual data analysis tool to assist domain experts to better understand functional safety datasets. Through a three-month-long user-centered design process comprising interviews, demos, and discussions with domain experts from a multinational automobile company, we derived design goals and user tasks that drove our work. After a final phase of user feedback on the prototype, SafetyLens is currently being deployed.

The primary contributions of this paper include (i) design goals and challenges in supporting visual analysis of Functional Safety datasets, (ii) SafetyLens, a visual data analysis tool that helps domain experts visualize and interact with functional safety datasets, (iii) usage scenarios that illustrate how SafetyLens can aid the current analysis processes of domain experts, and (iv) feedback from domain experts about the design, functionality, and impact of SafetyLens.

2 INDUSTRIAL BACKGROUND

Functional Safety analyses are performed in several domains such as Military Aviation [14], Space [49], Medical [32], Automotive [31], etc. Our work here is focused on Functional Safety of vehicles within the automotive systems engineering domain. ISO 26262 titled “Functional Safety - Road Vehicles” [31] is the international standard for the design and development of automotive electrical and electronic systems that makes Functional Safety a part of the automotive product development life-cycle. It is a risk-based safety standard, where the risk from hazardous situations is qualitatively assessed leading to the definition of safety measures that minimize these risks.

2.1 Concepts

This section introduces concepts and terminology associated with Functional Safety from the perspective of ISO 26262. These also describe the Functional Safety datasets used in SafetyLens. When modeled as a network, they make up the nodes, links, and attributes.

**SYSTEM** refers to a classification of automotive electrical and electronic systems such as “Adaptive Cruise Control” and “Chassis” on

| Element Type          | Abb. | Description                                                                 |
|-----------------------|------|-----------------------------------------------------------------------------|
| System Behavior       | SB   | An expected behavior of an electrical/electronic System in a vehicle.       |
| Malfunctioning Behavior | MB   | A failure or unintended behaviour of an item with respect to its design intent. |
| Situation             | Sit  | A vehicular operational scenario.                                           |
| Hazard                | Haz  | A potential source of harm.                                                 |
| Hazardous Event       | HzE  | A dangerous condition arising out of a Situation and a Hazard.               |
| Safety Goal           | SG   | A requirement to eliminate (or minimize) Hazards.                            |
| Functional Safety Requirement | FSR | A requirement to achieve the functional or behavioral aspects of a Safety Goal. |
| Technical Safety Requirement | TSR | A technical requirement to meet a Functional Safety Requirement.             |
| Software Safety Requirement | SSR | A software-level requirement.                                               |
| Hardware Safety Requirement | HSR | A hardware-level requirement.                                               |

Table 1. A description of the Types of Elements in the context of automotive Functional Safety. Other Element Types such as Physical, Logical, and Assumptions exist but are beyond the scope of this paper.

which Functional Safety is performed.

**PROJECT** refers to an instance of a System, e.g., “Chassis for Model 2”. This is also an instance of a Functional Safety dataset.

**ELEMENT** refers to an entity within a Project (e.g., “Vehicle should not accelerate more than required.”). Elements constitute nodes in a Functional Safety network.

**TYPE** refers to a group of similar Elements within a Project, e.g. Malfunctioning Behaviors (MB), described in Table 1. Type is an attribute of the nodes in Functional Safety networks.

**RELATION** refers to the relationship between two Elements. For example, a Relation “associatedSG” indicates that a Safety Goal (SG) has been assigned to a Hazardous Event (HzE). Relations constitute the links in Functional Safety networks. Some of these Relations are shown in the Links column in Figure 5.

**ASIL** Automotive Safety Integrity Level, or ASIL, is a nominal attribute assigned to an Element (node) indicating its relative risk level. It can take one of four values: {A, B, C, D}. An Element that is assigned an ASIL=D indicates the highest risk. ASIL=A indicates the lowest risk. A special value QM indicates that quality management processes are sufficient to manage the identified risk [31].

2.2 Workflow

Functional Safety is a broad process ranging from the design through to the manufacturing and quality management of vehicles. The processes relevant to our work here comprise the following:

**HAZARD ANALYSIS AND RISK ASSESSMENT (HARA):** Domain experts first determine potential hazards that can be triggered by the malfunctioning of one or more system components and/or processes. These hazards are classified based on their Severity (S), Exposure (E), and Controllability (C). Severity is an estimate of the extent of harm to one or more individuals that can occur in a potentially hazardous situation. Exposure is a measure of the probability of being in a hazardous situation. Controllability is an estimate of the driver’s ability to avoid harm or damage through timely reactions. These are qualitative ratings with Severity={S1, S2, S3}, Exposure={E0, E1, E2, E3, E4}, and Controllability={C0, C1, C2, C3}. Based on these, an Automotive Safety Integrity Level (ASIL) is calculated and assigned to the corresponding Hazardous Event (HzE). To minimize the risk associated with this hazard, a Safety Goal (SG) is formulated.

**FUNCTIONAL SAFETY CONCEPT (FSC):** From the formulated SG,
**Functional Safety Requirements (FSR)** are derived. These define a system architecture to achieve the SGS.

**Technical Safety Concept (TSC):** The technical safety concept comprises all Technical Safety Requirements (TSRs) along with their allocations to system, hardware or software elements. These are refinements of the corresponding FSRs.

In practice, Functional Safety analysts and engineers can use a combination of commercial and open source tools to perform their day-to-day tasks. The Functional Safety processes can either be performed and documented in such tools or using traditional approaches such as Excel sheets and Word documents. For instance, our domain collaborators have utilized a domain-specific language (DSL) for ISO 26262 that enables them to import artifacts from HARA, FSC, TSC into a model. They use this model to generate Word documents with information on functional safety that was performed on a system.

There can be several challenges in the above workflow, especially in the translation of an assigned ASIL into a Technical Safety Concept. Assigning an ASIL is a complex process that can involve discussions among multiple teams. Further, ASIL assignments can be modified at different functional safety stages and points in time. The introduction of model-based systems engineering (MBSE) approaches may solve part of the problem as collaborators can now exchange information using a common language like SysML [41]. However, this may still have limitations when the scale and complexity of data increases. Finally, analyzing multiple projects to detect shared components and problems is not supported, and hence, a core focus of SafetyLens.

### 2.3 Data

Each *Project* constitutes an instance of a Functional Safety dataset. As seen in Figure 2, this can be modeled as a network with nodes and links. Each node represents a functional safety *Element* with seven attributes: *(ID, Name, Type, ASIL, Severity, Exposure, Controllability).* A link between two nodes represents the *Relation* between two Elements and has three attributes: *(Source, Target, Relation).*

The dataset has several properties. First, a node of a particular Type can be connected to zero (orphan), one, or more nodes of some other Type. Second, Functional Safety Requirement (FSR) and Technical Safety Requirement (TSR) nodes can be connected to nodes of the same Type as themselves. Third, a dataset can comprise nodes and links that are common to other projects (e.g., the same hazard (HZE) can arise due to different malfunctioning system behaviors (MB)). Fourth, these common nodes may be assigned different ASIL values. Finally, a standard Functional Safety network may consist of multiple nodes and links with the vehicle itself having multiple systems. This scale and complexity makes Functional Safety datasets large, and make the tasks of our domain experts challenging.

### 3 Related Work

**Network Visualizations** Since a functional safety dataset can be modeled as a network, we explored existing literature in network visualization systems for inspiration. There are a number of open source, free, or commercially available software [5,8,9,13,50], toolkits [27,39], and research prototypes [33,48,54]. Multiple survey reports [11,28,55] have not only summarized the state-of-the-art of network visualizations and techniques but also discussed their general evolution. We review Saket et al.’s evaluation study that indicated node-link-group visualizations as more “enjoyable” than node-link visualizations [43] as a potential technique to represent the functional safety network. A core functional safety task is to compare projects (networks) hence we review Gove, R.’s V3SPA, a visual analysis tool (for security policy workers) to explore and find differences between large complex networks (SELinux and SEAndroid security policies) [26]. Another core functional safety task is to find and visualize connections and/or paths between elements (nodes) hence we also review existing work in route tracing techniques. Candela et al. have developed Radian, an internet probe that helps visualize route trace paths [17]. Fischer et al. have developed Vistracer to investigate routing anomalies in traceroutes [21]. Zhao et al have developed MissBin which infers the existence of unseen (missing) links based on currently observed ones by involving the user to sensemake the predicted results [57].

**Set Visualizations** Another functional safety task involves comparing graphs to find common nodes and links. For this, we explored existing literature in set visualization. Euler and Venn diagrams are the most common methods to visualize sets and their intersections. Euler diagrams represent each set as a geometric shape and show the intersections by overlapping the shapes. Venn diagrams are like Euler diagrams but show all intersections, including empty ones. Sadana et al. [42] developed Onset to represent large-scale binary set data. Lex et al.’s UpSet technique used a set view and element view to visualize intersections in a matrix layout introducing aggregates based on groupings and queries [36]. Alsallakh et al.’s survey paper discusses the state-of-the-art of set visualization systems and techniques [2].

**Visual Analytics in the Automotive Domain** Perhaps most relevant to this paper is previous work on visualization applied to the automotive industry. These had been mostly in the context of scientific visualization, computer aided design (CAD), and virtual reality [53]. Considerably less work was dedicated to the systems engineering domain until recently. Basole et al. developed visual analytics tools to help users perform Complex Engineered System (CES) design analysis tasks helping stakeholders with visualizations of complex design models [7] and understand Failure Mode Effect Analysis (FMEA) data by modelling and visualizing it as a network [6]. Sedlmair et al. have
developed multiple tools leveraging visual analytics in the electronic engineering domain for vehicle development and testing. One such tool is Cardiogram, that helps engineers debug millions of recorded messages from safety-critical in-car communication networks and ensure that they are error-free [47]. RelEx helps engineers specify and optimize traffic patterns for in-car communication networks [45]. MostVis facilitates exploration of MOST function catalogs which was otherwise infeasible using paper and existing database interfaces [44]. Another tool was a dual-view visualization system that helped diagnostics of in-car communication networks [46].

**Functional Safety** is a critical operation in automotive systems engineering whose methodologies [15, 30, 31] and applications [12, 18, 29, 56] have been extensively researched in literature. Several processes in systems engineering such as design reviews are often still conducted using unstructured documents (e.g., Excel sheets [37]) which is non-standard, inconsistent, and can cause miscommunication [22]. Part of the problem was solved by the introduction of model-based systems engineering (MBSE) approaches that encourage information exchange using a common language [41] (e.g., SysML [24]). There exist several tools now, both commercial [1.3] as well as open-source [23, 51] that support commonly performed tasks in systems engineering. However, these advances have resulted in more complex models and user tasks.

In general, these existing tools focus their user experience and functionality on the authoring of the information. These are valuable for creating the information, but lack user interface affordances for communicating and exploring the data for various tasks described in subsequent sections. Moreover, these applications cannot be customized to support newer or more advanced types of tasks. For example, comparing multiple projects or visualizing the ASIL traceability and decomposition cannot be easily accomplished in these tools. The potential application of viewing functional safety data in a visual analysis tool is under-explored, motivating the need for this work.

4 DESIGN PROCESS AND DOMAIN REQUIREMENTS

We adopted a user-centered design methodology and conducted a series of design activities to learn about Functional Safety, the requirements and tasks of the domain experts, and distill design goals that ultimately ground SafetyLens (see Figure 3). This process included formative evaluation, design sessions, and iterative prototype development.

First, we conducted interviews with functional safety engineers, analysts, and supervisors to understand the domain requirements and user tasks. We conducted these interviews remotely (using teleconference applications with video and screen sharing capabilities) over the course of three months, with one session per week. Generally, the sessions lasted 1 hour, with 1-4 domain experts on the calls. These sessions consisted of semi-structured interviews that started with questions to learn about their tasks, roles, and responsibilities. We gained an understanding of the current design ecosystem of tools, tasks, and datasets. We learned about the questions they have to answer as part of their daily activities, and specifically how those relate to the different datasets, databases, and other sources of information (e.g., manuals, regulations, etc.). We learned about the current workflow and identified 25 commonly performed tasks.

Second, we began design exercises to explore potential visualization and interaction techniques. We sketched ideas on paper and shared them with our collaborators during subsequent sessions. These sketches included potential visualization techniques (e.g., tree maps, force-directed networks, hive plots), UI widgets (e.g., sliders, dropdowns, buttons), UI layouts (e.g., panels, grids), workflows, and interactions. These were low-fidelity designs sketched quickly but detailed enough to catch errors and slips that could surface later. During these sessions, we collaboratively brainstormed on the pros and cons of each design which eventually resulted in multiple changes and refinements. We also realized these designs by developing rapid software prototypes with a dual purpose of exploring potential technologies for the tool (such as software libraries and packages) and evaluating the feasibility of the designs. We deployed these prototypes for the domain experts to interact and evaluate them based on their usability and visual look and feel, with small subsets of their data integrated into the prototypes. This helped discard less-useful designs, fine-tune our initial set of tasks, and derive 6 design goals.

Third, we designed and developed an initial prototype of SafetyLens and conducted an in-person demo and feedback session. This consisted of two half-day sessions with domain experts comprising functional safety analysts, project managers, and personnel from application-specific teams (e.g., chassis, power-train) who were split into two groups. The first session included 15 participants (7 in-person, 8 online), and the second one included 20 participants (6 in-person, 14 online). Each session consisted of a presentation of the design goals and supported user tasks, followed by a demonstration of the prototype. One researcher presented the prototype and led the discussion, while a second took notes on participants’ feedback. The participants engaged in thoughtful discussions about the demonstrated usage scenarios. They had access to sticky notes to keep track of comments throughout the presentation (with remote participants being able to use chat and other messaging services to give their feedback). Also, screen-shots were given to each participant to annotate with comments, changes, or other visual cues about how to improve the interface and interaction design.

Finally, a whiteboard was used by the in-person participants to illustrate how attributes from different databases can be integrated into SafetyLens. These discussions led to refinements to existing features, as well as new tasks that the tool could support. The user tasks and design goals presented below are those that resulted from this second round of user feedback.

4.1 User Tasks

User tasks which SafetyLens should support are:

- **Discover patterns within projects;** e.g., a project has elements that are mostly assigned an ASIL=D (high risk).
- **Find missing (otherwise must-have) links between nodes;** e.g., \{MB \rightarrow HzE\}, \{HzE \rightarrow SG\}, \{SG \rightarrow FSR\}, \{FSR \rightarrow TSR\}.
- **Look up and analyze a node’s end-to-end traceability (i.e., the extent to which functional safety elements have been defined);** e.g., \{MB \rightarrow HzE \rightarrow SG \rightarrow FSR \rightarrow TSR\}.
- **Compare ASILs by analyzing their decomposition into their S-E-C (Severity, Exposure, and Controllability) constituents.**
- **Find common nodes and links among projects;** (e.g., Hazards (Haz) that exist in multiple projects).
- **Discover anomalies across projects;** (e.g., the same Hazardous Event (HEE) is assigned a different ASIL across projects).
- **Compare key metrics across projects;** (e.g., counts and distribution of nodes and links within projects).
- **Get an overview of current and past projects to monitor status.**

4.2 Design Goals

From the design exercises and requirements gathering activities described above we identified the following design goals.
DG1: Facilitate Exploration of a Project User tasks such as “Find missing links”, “Lookup and analyze a nodes end-to-end traceability” are network exploration tasks. Hence, we modeled Functional Safety data as a network and derived this design goal to support tasks like fetching node details on hover, finding adjacent nodes, and finding paths from one node to another based on the taxonomies by Lee et al. [34] and Pretorius et al. [40].

DG2: Facilitate Comparison among Projects Teams within functional safety can be system-specific and may not always be aware of the day to day progress made by other teams. This may result in duplicate work (e.g., a team may re-implement an artifact from scratch instead of re-using the one already implemented by another team, or for another vehicle). A core goal for SafetyLens, thus, was to provide a unified interface where users can explore and compare multiple projects to find shared (common) nodes, links, or even sub-graphs (combination of nodes and links). This unified interface can foster better collaboration among teams while also saving time and resources for the organization.

DG3: Discover Patterns and Anomalies The domain experts we spoke to made use of existing commercial and open source tools. These tools support basic exploration and comparison of Functional Safety projects but may fall short when the scale and complexity of data increases. Thus, SafetyLens should support discovering interesting patterns within and across projects.

DG4: Facilitate Traceability and Decomposition of ASILs An important task for domain experts is to trace the ASIL from one node to another (e.g., \( MB \rightarrow HE \rightarrow SG \rightarrow FSR \rightarrow TSF \)). Since ASILs determine the extent of safety mechanisms for elements, any discrepancy such as an element assigned an ASIL=A instead of ASIL=D is an important concern. To diagnose the problem, users should be able to decompose the ASIL into its \( S \), Exposure (E), and Controllability (C). Thus, it is an important design goal for SafetyLens to unify tasks allowing users to efficiently detect, diagnose, and fix ASIL assignment issues.

DG5: Support Different User Groups Functional Safety consists of engineers, analysts, and managers all of whom perform analysis and decision-making tasks at different levels. While engineers and analysts are concerned with lower element-level tasks within projects, managers are concerned with higher project-level tasks. SafetyLens should support both along the functional safety organization hierarchy.

DG6: Provide a Summary of Key Metrics The user tasks suggested that key metrics should be readily available (e.g., total number of nodes, total number of nodes with ASIL=D, etc). In addition to these design goals, we had to consider other factors. Since Functional Safety datasets can get large, we had to design for scalability and performance. We utilized a graph database to persist the data and execute queries to offload computation on the browser. Further, we had to ensure our tool is easy-to-deploy and integrates well with existing systems and workflows (e.g., the use of linked spreadsheets that are local to groups and users as well as shared databases). Finally, our users are from the automotive domain and not visualization practitioners, hence SafetyLens needs a simple user experience.

4.3 Design Considerations Since we modeled Functional Safety data as a network we considered several network visualization techniques. We implemented rapid prototype standard visualization techniques such as hive-plots, dot-matrix plots, and parallel coordinate charts. These had a number of shortcomings. First, due to the number of links between nodes, edge-crossings can get messy. Second, positioning the nodes linearly in hive-plots made it difficult to discover patterns (e.g., distribution of ASILs). The dot-matrix plot enabled discovering patterns but give an impression that the \( \{x,y\} \) spatial coordinates of each node are of importance which was not the case. Third, parallel coordinates charts failed because the nodes had connections with more than two types of nodes.

We implemented a force-directed network with a multi-body force algorithm generating node positions at random. However, similar nodes (e.g., Elements with the same Type) were scattered throughout the canvas making pattern discovery difficult. Inspired by Saket et al’s study that found node-link-group diagrams more “enjoyable” than node-link diagrams [43], we augmented our visualization with circle packing (to group similar nodes together within a cluster), and collision detection (to prevent overlaps between clusters and nodes) algorithms to achieve our goal. To facilitate interactive exploration and “fluidity” [20], we provide affordances to drag the clusters within the canvas, brush and link between projects by a lasso operation, and a control panel with search and filter direct manipulation widgets.

To support comparison between projects, we show multiple node-link-group diagrams aligned spatially. One idea was to have a single visualization with all projects on a shared canvas. However, this had a major shortcoming. Since projects comprise shared nodes with attributes that are specific to a project (e.g., Node A in Project A has an ASIL=B but ASIL=D in Project B), it was important to keep the nodes and hence the projects separate. Our users are familiar with tabular resources for the organization.

5 SafetyLens

This section describes the main views and functionality of SafetyLens.

5.1 User Interface

SafetyLens has three primary views: 1) Dashboard View, 2) Summary View, and 3) Compare View.

Dashboard View, shown in Figure 4, is the landing page and provides an overview of the Functional Safety ecosystem. It consists of a clustered node visualization with each node representing a project. We complemented the visualization with a table to provide a familiar interface to users who are more comfortable with data in spreadsheets. The table consists of project-level information such as \{Name, Department, Project In-Charge, and Location\}. SafetyLens supports multiple starting points for users to analyze functional safety data. By providing access to all projects within the organization, SafetyLens facilitates collaboration (DG2) among domain experts with different roles (DG5).

Summary View, shown in Figure 5, provides a summary of the projects selected in the Dashboard View (DG6). There are juxtaposed heatmap visualizations for Node Type, Link Relation, and ASIL respectively positioned next to each other. For each attribute table, projects are along the column axis and the corresponding attribute values are along the row axis. An additional column titled “S” is added to show the number of entities (nodes and links) that are shared among these.
The projects share list enumerations match with those in Figure 1): analysis across multiple projects. It has four subviews (alphanumeric exploratory analysis of an individual project as well as a comparative helping the user discover patterns within as well as across projects (DG3).

Visual clusters leading to discoveries of patterns and anomalies (ii) export the panel as an image, (iii) toggle full-screen mode, (iv) toggle full-screen mode, (v) drag and drop. The visualization nodes and the datatable rows can be selected by their names.

SAFETYLENS first checks if a path exists between the two and another as traced their ASILs and visualize if a path exists between two nodes as well. Chris and their colleagues have been actively working on Project-A (the “Adaptive Cruise Control” System) for a week. They have (i) identified several Malfunctioning Behaviors and the potential Hazards that could arise, (ii) assigned a risk level (ASIL) to these hazards and (iii) defined Safety Goals and requirements (Functional Safety Requirements) to minimize these risks. Chris wishes to perform a status nodes and overlays it onto the node-link group visualization. It also returns a linearized node-link diagram showing the entire route from source to destination. Below the node-link diagram is a heatmap showing the S-E-C (Severity-Exposure-Controllability) break up of the ASILs.

Since the visualization canvas and the tab layout are vertically stacked within a project panel, we positioned each panel side by side. This way, SafetyLens would facilitate exploratory analyses within and comparative analyses across projects (DG1, DG2).

(B) SHARED VIEW shows the nodes and links that are common to / shared by all loaded projects (DG2). These are computed by performing a set intersection operation across project graphs based on unique node identifiers. We show it in a separate view instead of overlaying or highlighting in the same view to make it easier for domain experts to begin their analysis.

(C) DETAILS VIEW shows all attributes of a node as well as a table containing all nodes that are connected to this node.

(D) CONTROL PANEL has three tabs: (i) Nodes, (ii) Links, and (iii) Config with various operations that can be performed (e.g., lookups, encodings, layout, preferences for tooltips, and more).

5.2 Interactions

The primary design goals for the user interactions are to facilitate brushing and linking between views while maintaining a simple and usable interface. These interaction include:

(A) HOVER The visualization nodes and the datatable rows can be hovered to highlight neighbours as well as show more information in the Details View.

(B) CLICK The visualization nodes and the datatable rows can be (left) clicked to toggle their selections. In addition, right-clicking on a visualization node opens a context-menu that supports operations such as Select, Set as Source/Destination, and others.

(C) DRAG The group nodes can be dragged to move the clusters within the visualization canvas. Similarly, the project panels themselves can be reordered to aid comparison.

(D) SELECTION The visualization canvas can be used to draw freeform shapes via a lasso operation to select nodes and links.

(E) SEARCH The Search input field in the Control Panel can be used to “lookup” and “select” nodes by their names.

(F) FILTER The Control Panel has UI controls (e.g., radio buttons, checkboxes, dropdowns, interactive legends) that allow the user to “filter” and “select” the nodes and links in the visualization.

(G) SORT The column headers of datatables in the Tab layout can be used to sort the selected nodes and links.

5.3 Implementation

SafetyLens is implemented as a web application using Flask (a Python-based microframework) [4], AngularJS [25], and D3.js [38]. The Functional Safety data is stored in OrientDB [16].

6 Usage Scenarios

We illustrate how SafetyLens can help domain experts visualize and interact with Functional Safety datasets. Consider three usage scenarios comprising several subtasks as performed by three hypothetical users - Chris (analyst), Parker (Chris’s supervisor), and Kyle (analyst) respectively. These usage scenarios were developed in collaboration with the domain experts from our interview studies to ensure domain relevance. Due to confidentiality concerns, project, node, and link names have been obfuscated. These scenarios are also illustrated in the accompanying video.

6.1 General Exploratory Tasks
check to determine if the risks associated with the project components have been reduced to acceptable levels. This includes the following steps (or tasks).

**Find and address Orphan Nodes** It is important to ensure that the defined nodes are connected to other nodes (that is, are not orphan). By being aware of these orphan nodes, Chris can ensure that these are not implemented from scratch but instead reused from the same node’s implementation in another project.

To accomplish this task, Chris first opens the Control Panel and drags the Degree slider to zero. This filters out the nodes that have at least one connection. Using a lasso operation, they select these nodes and export a snapshot of the visualization for future reference. Figure 6 summarizes this task.

**Find and report nodes with an unassigned ASIL.** After addressing orphan nodes, Chris shifts their focus to assigning an ASIL to each node. The ASIL determines the quality and quantity of safety measures that are undertaken to minimize associated risks. During this assignment process, it is possible that a few nodes are either missed or skipped for later. In both cases, the node is left without a valid ASIL.

To verify this for **Hazardous Events**, Chris opens the Control Panel and hides all node types except **Hazardous Event**. From the ASIL legend, they select the nodes with an ASIL “.” (unassigned). Several nodes (gray) that do not have an ASIL are selected and added to the datatable below. Chris shares the selected nodes with their team to discuss and plan next steps. Chris’s workflow is summarized in Figure 7.

**Find Missing Links.** After analyzing orphan and ASIL-unassigned nodes, Chris funnels their attention to analyze specific connections within the Functional Safety network to ensure that:

1. Each **Malfunctioning Behavior (MB)** should have a **Hazardous Event (HzE)** identified for it.
2. Each **HzE** should have a **Safety Goal (SG)** assigned to it.
3. Each **SG** should define a **Functional Safety Requirement (FSR)**.
4. Each **FSR** should define a **Technical Safety Requirement (TSR)**.

A domain expert’s role is to address such missing connections at the earliest. The links in the dataset comprise the **Relation attribute** that determines the type of connection between two nodes, e.g., **associatedHE** connects a **Malfunctioning Behavior** with a **Hazardous Event**. A unique Relation is defined for a connection between nodes of different Types. To find these missing connections, Chris opens the Control Panel and switches to the Links tab. As illustrated in Figure 8, they select the corresponding Relations from the legend that are highlighted in the visualization. Chris observes that:

**6.2 Trace ASIL**

Another aspect of an analyst’s role is to analyze the ASIL of a node in comparison with the ASILs of its connections (neighbor, neighbor’s neighbor, and so on). Consider a **Hazardous Event (HzE)** that is assigned the highest risk classification ASIL-D. It requires **Safety Goals (SG)** that can sufficiently minimize the associated risk. By default, **SGs** inherit the ASIL of the **HzE**. **FSRs** inherit the ASIL of the **SGs**, and **TSRs** inherit the ASIL of the **FSRs** (associated with a selected **HzE**). The analyst may choose to override it to achieve acceptable risk levels, making visual analysis of the ASIL trace an important task.

Kyle’s role is to assess risks associated with **HzE**s and qualitatively assign an ASIL to them. Their responsibilities also include analyzing the extent to which Functional Safety is achieved for a node, that is, finding paths between nodes that are not directly connected via a link. This way, a project’s maturity (“Are there sufficient nodes and links implemented for this project?”) as well as completeness (“Most SBs have a TSR defined now; the project is nearing its completion.”) can be assessed. Kyle performs the following tasks:

**Manually Trace paths.** As illustrated in Figure 9, Kyle chooses a known **SB** for further analysis. (1) On hovering, they see several links connecting it with multiple **MBs**. (2) They select the **SB** and hover over one of the **MB**s which highlights the links with **HzEs**. (3,4) They continue this process until they reach an **FSR**. (5) They find that the **FSR** does not have a link with an **TSR**. Since this is a requirement, this means Functional Safety is still incomplete for the **SB** in (1).

**Find if a Path exists between two Nodes.** In the previous task, Kyle traversed the network manually by following the nodes’ connections. SafetyLens can also check if a path exists between non-adjacent nodes,
that is, nodes that are not directly connected. Consider a scenario to check if a Technical Safety Requirement (TSR) exists for a System Behavior (SB) as illustrated in Figure 11. Kyle imports Project-A into the Compare View and switches to the Trace Tab in the Project View. (1) They right click on the SB and set it as Source from the context menu. (2) They look up the TSR from the search field and set it as Destination. (3) SafetyLens calculates and finds that a path does exist between the two nodes. In this way, SafetyLens helps analysts find and visualize paths between two nodes.

**Compare ASILs along a Path.** Figure 11 also illustrates an extension of the previous task to analyze the ASILs of intermediary nodes along the path. (4) The node-link (rectangles) diagram below reshapes the overlaid path in the visualization into a linear layout with the Source and Destination nodes at the ends. The rectangle’s fill color is determined by the ASIL of the node. Kyle observes that the same ASIL C has propagated from the Hazardous Event (HzE) to the Technical Safety Requirement (TSR). To verify if the ASILs are really consistent, they need to analyze their respective breakups into Severity (S), Exposure (E), and Controllability (C) values. (5) SafetyLens shows this decomposition of ASIL in the form of a heatmap. Kyle observes that while \{N92, N136, N187, N209\} have consistent ASILs, they do not have consistent Severity (S), Exposure (E), and Controllability (C). For example, the Controllability of N136 is C2 whereas it should have ideally been C3 (from N92). In this way, SafetyLens helps analysts trace ASILs within the functional safety network.

**6.3 Find Common Elements Between Projects**

Until now we have illustrated how SafetyLens helps domain experts explore and analyze one project at a time. SafetyLens also supports simultaneous comparative analysis among multiple projects. We illustrate this with another scenario.

In Section 6.1, we illustrated how SafetyLens helped Chris find and report Malfunctioning Behaviors (MB) that did not have a corresponding Hazardous Event (HzE). With resources likely already allocated for this project, Chris showed the list to his supervisor, Parker. On seeing the list, Parker is able to recall a previous project Project-C where the same elements may have already been defined. They open SafetyLens but this time select two projects: Project-A and Project-C from the Dashboard view. An overview (distributions and counts) of these projects including shared nodes and links can be seen from the Summary View similar to that shown in Figure 5. Parker sees that there are common nodes and links among these projects and switches to the Compare View to find the specific ones.

As illustrated in Figure 10, they see a new Shared Nodes panel at the top left. SafetyLens, by applying set intersection between network nodes and links, has pre-computed the shared nodes and links between the two projects and presented in the same node-link group visualization. They hover on a Malfunctioning Behavior (MB) and find that it is in fact on Chris’s list. (1) Hovering on this node in the shared view highlights the corresponding nodes in Project-A and Project-C along with their connections, if any. While there are rightly no links from
ASIL C has undergone revisions, e.g. was previously D?" This opened up an entire aspect of visualizing temporal changes to data. This was performed remotely, with our collaborator at the company.

7.1 Feedback

7.2 Deployment

7.3 Limitations

8 Conclusion

Automotive domain experts perform functional safety of the vehicles to minimize risks associated with hazards. These domain experts currently spend a significant amount of time gathering and analyzing data. We designed and developed SafetyLens, a visual data analysis tool to help them visualize and interact with Functional Safety datasets. SafetyLens is a tool that aims to assist Functional Safety analysis by showing relationships and patterns across projects. This paper explains our design process, how the user tasks and design goals guided SafetyLens’s interface design and development, and provided usage scenarios to explain how the tool can be used to perform existing as well newer advanced analysis tasks.

References

[1] No Magic, 2014.
[2] B. Alsallakh, L. Micallef, W. Aigner, H. Hauser, S. Miksch, and P. Rodgers. The state-of-the-art of set visualization. In Computer Graphics Forum, volume 35, pages 234–260. Wiley Online Library, 2016.
D. Ward, I. Ibarra, and A. Ruddle. Threat analysis and risk assessment in automotive cyber security. *SAE International Journal of Passenger Cars-Electronic and Electrical Systems*, 6(2013-01-1415):507–513, 2013.

J. Zhao, M. Sun, F. Chen, and P. Chiu. Missbin: Visual analysis of missing links in bipartite networks. In *2019 IEEE Visualization Conference (VIS)*, pages 71–75, 2019.